Improved Iterative Schemes for REML Estimation of Variance Parameters in Linear Mixed Models

Emma Knight

School of Agriculture, Food and Wine
The University of Adelaide

13 October 2008
# Contents

Abstract vii  
Declaration ix  
Acknowledgements xi  

1 Introduction 1  
1.1 Review of comparisons of algorithms used to compute REML estimates of variance parameters 3  
1.2 Methods used to calculate estimates of variance parameters in common statistical software 4  
1.3 Improved Iterative Schemes 6  
1.4 Outline 7  

2 Mixed Models 9  
2.1 Model 9  
2.2 Estimation 11  
2.2.1 Best Linear Unbiased Prediction 11  
2.2.2 REML Estimation of Variance Parameters 17  
2.3 Iterative Schemes 27  

3 AI Algorithm 29  
3.1 Introduction 29  
3.2 AI Algorithm 33  
3.3 Absorption, Backsubstitution and Inversion 35  

4 EM Algorithm 41  
4.1 Introduction 41  
4.2 Definition of the EM Algorithm 45  
4.3 Monotonicity of the EM Algorithm 45  
4.4 Self-Consistency of the EM Algorithm 47
4.5 REML EM Algorithm for Mixed Models ........................................ 48

5 PXEM Algorithm 55
5.1 Introduction ........................................................................ 55
5.2 REML PXEM Algorithm for Mixed Models ......................... 56

6 Rates of Convergence 67
6.1 Introduction ........................................................................ 67
6.2 Convergence Criterion ....................................................... 67
6.3 Order of Convergence ......................................................... 68
   6.3.1 Newton-Raphson Algorithm .......................................... 69
   6.3.2 Fisher Scoring Algorithm .............................................. 70
   6.3.3 Average Information Algorithm .................................... 71
   6.3.4 EM Algorithm ............................................................ 71
   6.3.5 PXEM Algorithm ....................................................... 76
   6.3.6 Measuring the Linear Rate of Convergence .................... 78

7 Improved Iterative Schemes 81
7.1 Introduction ........................................................................ 81
7.2 Hybrid Schemes ................................................................ 81
7.3 Local Schemes .................................................................. 83
   7.3.1 Local EM Algorithm ................................................. 87
   7.3.2 Local PXEM Algorithm ............................................. 88
   7.3.3 Implementation of Local Schemes ................................. 94
7.4 Starting Values .................................................................. 95

8 Variance Components Model 97
8.1 Introduction ....................................................................... 97
8.2 Variance Components Model and Results .......................... 97
   8.2.1 AI Algorithm .......................................................... 98
   8.2.2 EM Algorithm ........................................................ 99
   8.2.3 PXEM Algorithm .................................................... 101
   8.2.4 Local EM Scheme .................................................. 106
   8.2.5 Local PXEM Scheme .............................................. 107
   8.2.6 Starting Values ....................................................... 108
8.3 Lamb Weight Data ............................................................ 111
8.4 Simulated Incomplete Block Design Data ......................... 118
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.4.1</td>
<td>Incomplete Block Design with 9 replicates</td>
<td>119</td>
</tr>
<tr>
<td>8.4.2</td>
<td>Incomplete Block Design with 3 replicates</td>
<td>135</td>
</tr>
<tr>
<td>8.5</td>
<td>Robust Iterative Scheme</td>
<td>150</td>
</tr>
<tr>
<td>9</td>
<td><strong>Unstructured G Model</strong></td>
<td>157</td>
</tr>
<tr>
<td>9.1</td>
<td>Introduction</td>
<td>157</td>
</tr>
<tr>
<td>9.2</td>
<td>Unstructured G Model and Results</td>
<td>158</td>
</tr>
<tr>
<td>9.2.1</td>
<td>AI Algorithm</td>
<td>161</td>
</tr>
<tr>
<td>9.2.2</td>
<td>EM Algorithm</td>
<td>162</td>
</tr>
<tr>
<td>9.2.3</td>
<td>Computing EM updates</td>
<td>167</td>
</tr>
<tr>
<td>9.2.4</td>
<td>PXEM Algorithm</td>
<td>169</td>
</tr>
<tr>
<td>9.2.5</td>
<td>Local EM Scheme</td>
<td>175</td>
</tr>
<tr>
<td>9.2.6</td>
<td>Local PXEM Scheme</td>
<td>177</td>
</tr>
<tr>
<td>9.2.7</td>
<td>Starting Values</td>
<td>178</td>
</tr>
<tr>
<td>9.2.7.1</td>
<td>Multivariate Data</td>
<td>178</td>
</tr>
<tr>
<td>9.2.7.2</td>
<td>Random Coefficient Data</td>
<td>184</td>
</tr>
<tr>
<td>9.3</td>
<td>Random Coefficient Data Sets</td>
<td>186</td>
</tr>
<tr>
<td>9.3.1</td>
<td>Orthodontic Data</td>
<td>186</td>
</tr>
<tr>
<td>9.3.2</td>
<td>Ultrafiltration Data</td>
<td>196</td>
</tr>
<tr>
<td>9.3.3</td>
<td>Simulated Random Coefficient Data</td>
<td>206</td>
</tr>
<tr>
<td>9.4</td>
<td>Multi-Environment Plant Variety Data</td>
<td>229</td>
</tr>
<tr>
<td>9.4.1</td>
<td>Simulated Multi-Environment Plant Variety Data</td>
<td>229</td>
</tr>
<tr>
<td>9.5</td>
<td>Robust Iterative Scheme</td>
<td>260</td>
</tr>
<tr>
<td>10</td>
<td><strong>Conclusion</strong></td>
<td>267</td>
</tr>
<tr>
<td>Appendix</td>
<td></td>
<td>273</td>
</tr>
<tr>
<td>Bibliography</td>
<td></td>
<td>283</td>
</tr>
</tbody>
</table>
Abstract

Residual maximum likelihood (REML) estimation is a popular method of estimation for variance parameters in linear mixed models, which typically requires an iterative scheme. The aim of this thesis is to review several popular iterative schemes and to develop an improved iterative strategy that will work for a wide class of models.

The average information (AI) algorithm is a computationally convenient and efficient algorithm to use when starting values are in the neighbourhood of the REML solution. However when reasonable starting values are not available, the algorithm can fail to converge. The expectation-maximisation (EM) algorithm and the parameter expanded EM (PXEM) algorithm are good alternatives in these situations but they can be very slow to converge. The formulation of these algorithms for a general linear mixed model is presented, along with their convergence properties.

A series of hybrid algorithms are presented. EM or PXEM iterations are used initially to obtain variance parameter estimates that are in the neighbourhood of the REML solution, and then AI iterations are used to ensure rapid convergence. Composite local EM/AI and local PXEM/AI schemes are also developed; the local EM and local PXEM algorithms update only the random effect variance parameters, with the estimates of the residual error variance parameters held fixed. Techniques for determining when to use EM-type iterations and when to switch to AI iterations are investigated. Methods for obtaining starting values for the iterative schemes are also presented.

The performance of these various schemes is investigated for several different linear mixed models. A number of data sets are used, including published data sets and simulated data. The performance of the basic algorithms is compared to that of the various hybrid algorithms, using both uninformed and informed starting values. The theoretical and empirical convergence rates are calculated and compared for the basic algorithms.

The direct comparison of the AI and PXEM algorithms shows that the PXEM algorithm, although an improvement over the EM algorithm, still falls well short of the AI algorithm in terms of speed of convergence. However, when the starting values are too far from the REML solution, the AI algorithm can be unstable. Instability is most likely to arise in models with a more complex variance structure. The hybrid schemes use EM-type iterations to move close enough to the REML solution to enable the AI algorithm to successfully converge. They are shown to be robust to choice of starting values like the EM and PXEM algorithms, while demonstrating fast convergence like the AI algorithm.
Declaration

This work contains no material which has been accepted for the award of any other degree or diploma in any university or other tertiary institution and, to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made in the text.

I consent to this copy of my thesis, when deposited in the University Library, being made available for loan and photocopying, subject to the provisions of the Copyright Act 1968.

Emma Knight
Acknowledgements

Thankyou to my supervisors Professor Ari Verbyla, Professor Brian Cullis and Professor Robin Thompson for their guidance, support and encouragement.

I thank my colleagues at the Australian National University for their continual support and encouragement; Jeff Wood for providing a sounding board and for reading and commenting on sections of my thesis, Ross Cunningham for letting me know that the tough days towards the end were not unusual, Christine Donnelly for the pep talks that helped to motivate and focus me, David Lindenmayer for his patience and understanding, and to Rachel Muntz, Sam Banks, Adam Felton and Diane Jakobasch for listening when I just needed someone to listen.

To those friends who were brave enough to enquire about the progress I was or was not making, I thank them for their best wishes and for putting up with my sometimes unfriendly responses. And to those friends who knew not to ask, thankyou. In particular I would like to thank Sheridan Price, Luke Cartwright, Bec Darbyshire, Jen and Sean Moran and Damien Brown for their friendship, support and encouragement.

I am grateful to my family for their continual love and support, and for their belief that I would eventually get there. Finally I thank my husband Paul for his endless patience, understanding and love, and for not only providing, but also hiding and rationing chocolate supplies.
Chapter 1

Introduction

Linear mixed models are a powerful class of models used for the analysis of correlated data. Examples of correlated data that may be appropriately analysed using a linear mixed model include studies with grouped or clustered data and repeated-measures or longitudinal data. Grouped or clustered data occur when observational units are grouped into or nested within clusters of units. For example data from students in classrooms, siblings within families, patients within hospitals and block and split-plot designs. Repeated-measures data arise when repeated observations are taken on the same observational unit, either over time or under changing experimental or observational conditions.

Observations from the same group or repeated observations on the same observational unit are not independent. By associating common random effects to observations within the same group, mixed models are able to specify the covariance structure induced by the grouping of the data. A wide range of covariance models are possible, making the mixed model a very flexible tool for analysing correlated data. As a result, the linear mixed model has become a widely used tool for data analysis in a range of applications including the agricultural, environmental, biological, medical and social sciences.

The popularity of linear mixed models has rapidly expanded in recent years due to an increase in the availability of software as well as a number of text books that provide an overview of the theory and application of linear mixed models. These include Searle et al. (1992), Verbeke and Molenberghs (2000), Pinheiro and Bates (2000), McCulloch and Searle (2001), Cox and Solomon (2003), Demidenko (2004), Rabe-Hesketh and Skrondal (2005), Galwey (2006), Brown and Prescott (2006), West et al. (2006) and Jiang (2007).

Estimation in a linear mixed model involves two linked processes: estimation/prediction of the fixed and random effects, and estimation of the variance parameters. The fixed and random effects are estimated/predicted using the method first established by Henderson (1950) and Henderson (1963) which later became known as best linear unbiased prediction. Several methods have been used to estimate variance parameters in linear mixed models. Two commonly used methods are maximum likelihood (ML) estimation (Hartley and Rao, 1967) and residual maximum likelihood (REML) estimation (Patterson and Thompson, 1971). REML is the preferred and most common method of estimation for variance parameters in linear mixed models as it corrects for the bias in ML estimation.
by accounting for the loss of degrees of freedom attributable to estimation of the fixed effects.

In linear mixed models, the REML log-likelihood is a non-linear function of the variance parameters. Maximisation of the REML log-likelihood to obtain estimates of the variance parameters typically involves the use of an iterative scheme. A wide variety of iterative techniques have been proposed based on various modifications of two basic approaches: the Expectation-Maximisation (EM) algorithm and the Newton-Raphson (NR) algorithm.

The NR algorithm is one of the most widely used optimization procedures in statistical computation. It is a second order algorithm and hence enjoys quadratic local convergence. In their original description of REML estimation, Patterson and Thompson (1971) proposed the use of the Fisher Scoring (FS) algorithm. The FS algorithm differs from the NR algorithm by using the expected information matrix in place of the observed information matrix. Harville (1977) mentions that the advantage of the FS algorithm over the NR algorithm is that, since the expected information matrix is often easier to compute than the observed information matrix, it will generally require less computer time per iteration, though possibly at the expense of an increased number of iterations to convergence.

The observed information and expected information matrices involve traces of products of large matrices which can be computationally intensive. To overcome this problem, Johnson and Thompson (1995) and Gilmour et al. (1995) proposed the use of the Average Information (AI) algorithm which averages the observed and expected information matrices, canceling out these trace terms. The AI algorithm is a computationally convenient and efficient algorithm to use when the starting values for the variance parameters are within the neighbourhood of the REML solution. However, when reasonable starting values are not available, the algorithm can fail to converge. Like all NR type algorithms, the AI algorithm does not guarantee an increase in the log-likelihood with each iteration, nor does it guarantee to keep the variance parameters within the parameter space. In the neighbourhood of the REML solution, the AI algorithm generally yields updates that are within the parameter space and results in an increase in the log-likelihood with each iteration. However, when poor starting values are used, the AI algorithm is prone to overshooting, resulting in variance parameter estimates that are outside the parameter space and/or a decrease in the log-likelihood.

The EM algorithm (Dempster et al., 1977) is a good alternative in these situations as it is more robust to starting values than NR type algorithms; the variance parameters are guaranteed to remain in the parameter space and the log-likelihood does not decrease after an EM iteration. However the EM algorithm can be very slow to converge. The variance parameter estimates rapidly approach the REML solution in the early iterations but then can converge very slowly in the neighbourhood of the REML solution. Many extensions and variations of the EM algorithm have been proposed to overcome the problem of slow convergence. Liu et al. (1998) presented the Parameter Expanded EM (PXEM) algorithm that has a faster rate of convergence while still maintaining the stability of the EM algorithm.
1.0. REVIEW OF COMPARISONS OF ALGORITHMS

The aim of this research is to evaluate and compare the performance of these various algorithms for a range of models using a range of data sets, and to build on these algorithms to develop an improved iterative scheme that performs well for a wide class of models. Before doing so, let us review some of the previous work that has compared iterative schemes for REML estimation of variance parameters in linear mixed models, and the methods implemented in some common statistical software.

1.1 Review of comparisons of algorithms used to compute REML estimates of variance parameters

Dempster et al. (1984) compared the EM and NR algorithms for a simple variance components model for two data sets and found that the NR algorithm converges in fewer iterations than the EM algorithm. They commented that the EM algorithm makes greater progress in the first few iterations, while the NR algorithm converges quickly once estimates are close to the solution. Lindstrom and Bates (1988) also compared the EM and NR algorithms for two repeated measures data sets and concluded that a well-implemented NR algorithm is preferable to the EM algorithm. They commented that the EM algorithm has been preferred in the past because each iteration could be computed more quickly. However they showed that if the number of random effects is relatively small and the NR is implemented well, there is no significant computational penalty in using the NR algorithm.

Harville and Callanan (1991) and Callanan and Harville (1991) compared the NR, FS and EM algorithms for a simple variance components model with a single random effect. They used a number of different parameterisations for each of the algorithms. They discuss the importance of determining the computational burden imposed by these algorithms not only by the number of iterations required to attain convergence but also by the extensiveness of the computations to carry out each iteration. They found that the fastest convergence (as measured by the number of iterations) was achieved by the NR algorithm, followed by the FS algorithm. The EM algorithm was very slow to converge. They found that for the example they looked at, the computing costs for all algorithms were small, with the NR and FS having slightly larger computing costs than EM. However they comment that in more extensive applications, the NR and FS algorithms are likely to have significantly greater computing costs than the EM algorithm.

Graser et al. (1987) extended the work of Smith and Graser (1986) and presented a derivative free (DF) algorithm which obtains estimates of the variance parameters by explicitly evaluating the likelihood and maximising it by direct search. Johnson and Thompson (1995) compared the AI algorithm to the DF and EM algorithms for a univariate animal model (variance components model) and found that the AI algorithm was about five times faster than the DF algorithm, which was about three times faster than the EM algorithm. Gilmour et al. (1995) compared the AI algorithm to the DF and EM algorithms for the analysis of data from a wheat variety trial, and found that the AI algorithm outperformed the DF and EM algorithms. They also compared the AI algorithm to the FS algorithm for an incomplete block analysis and for spatial models of field experiments and found that
the convergence sequences were similar for a range of models and data sets. Hence the AI algorithm appears to have similar convergence properties to the FS algorithm while avoiding the computational difficulties associated with evaluating the traces of large matrices in the computation of the expected information matrix. Meyer and Smith (1996) compared the DF algorithm to the Newton-type algorithms (including the AI algorithm) and found that the DF algorithm performed favourably compared to the Newton-type algorithms for univariate animal models. However, they found that the convergence rate of the DF algorithm declined with an increase in the number of variance components, with Newton-type algorithms performing better than derivative-free procedures for multivariate animal models (unstructured random effects variance matrix), the more so the larger the number of parameters to be estimated.

Foulley and van Dyk (2000) compared the performance of the EM and PXEM algorithms and found much better results in terms of convergence characteristics (number of iterations and time required for convergence) were obtained for the PXEM algorithm relative to the EM algorithm for a random regression example. Cullis et al. (2004), Meyer (2006a) and Meyer (2008) compared the EM and PXEM algorithms to the AI algorithm. Cullis et al. (2004) found that for a variance components and random coefficients model, the PXEM algorithm, although an improvement over EM, still fell well short of the AI algorithm in terms of number of iterations to convergence. However, with poor starting values, they found the AI algorithm was unstable. Cullis et al. (2004) also looked at hybrid EM/AI and PXEM/AI schemes and found the hybrid schemes outperformed the EM and PXEM algorithms. Meyer (2006a) and Meyer (2008) found that the AI algorithm outperformed the EM and PXEM algorithms, however the AI algorithm was sensitive to choice of starting values. Meyer (2006a) found the performance of the PXEM algorithm in the first few iterations was generally at least as good as that of the AI algorithm for multivariate animal models. However, in later iterations, PXEM shared the slow convergence of the EM algorithm. Using a few PXEM iterations then switching to AI iterations proved highly effective due to the PXEM algorithm yielding estimates sufficiently close to REML solution to allow the AI algorithm to converge rapidly, even if the AI algorithm had failed for the same starting values.

1.2 Methods used to calculate estimates of variance parameters in common statistical software

The MIXED procedure in SAS (SAS Institute Inc., 1999) fits linear mixed models using REML with a Newton-Raphson algorithm. The MIXED procedure has three options for the method of estimation. They are: ML and REML (the default method), which are both iterative methods and MIVQUE (Minimum Variance Quadratic Unbiased Estimation) which is a non-iterative method (Rao, 1972, LaMotte, 1973, Wolfinger et al., 1994). REML and ML estimation methods are implemented with a Newton-Raphson algorithm. Note that by default the MIXED procedure uses MIVQUE estimates as starting values for the ML and REML procedures. In the documentation for the MIXED procedure it is noted that the Newton-Raphson algorithm can fail to converge for some data sets. Convergence problems may be avoided by using the Fisher scoring algorithm rather than the default
1.2. METHODS USED IN COMMON STATISTICAL SOFTWARE

Newton Raphson algorithm. It is also possible to start the iteration process with a certain number of Fisher scoring iterations before switching to Newton Raphson iterations. A wide variety of variance models can be fitted using the MIXED procedure in SAS including scaled identity, compound symmetry, autoregressive and unstructured variance matrices. Littell et al. (1996) and Verbeke and Molenberghs (1997) give an overview of fitting linear mixed models in SAS.

In SPSS (SPSS Inc., 2002), linear mixed models can be fitted using either ML or REML (default) with either a Fisher Scoring or Newton Raphson algorithm following the methods of Wolfinger et al. (1994). The supporting documentation (SPSS Inc., 2002) states that the default is to use the Fisher scoring algorithm for the first iteration because it is more robust to poor initial values, following Jennrich and Sampson (1976). For subsequent iterations, the Newton-Raphson algorithm is used if the Hessian matrix is nonnegative definite, and the increment in the log-likelihood function of step 1 is less than or equal to one. Otherwise the Fisher scoring step is used. A similar range of variance models can be fitted in SPSS as in SAS, including scaled identity, compound symmetry, autoregressive and unstructured variance matrices. West et al. (2006) give an overview of fitting linear mixed models in SPSS (as well as SAS and R).

The lme function in the nlme package (Pinheiro et al., 2007) in S-PLUS (MathSoft, Inc, 1999) and R (R Development Core Team, 2007), and the more recent lmer function in the lme4 package (Bates, 2007b) in R, fit linear mixed effects models using either ML or REML with REML as the default (Pinheiro and Bates, 2000). The lme and lmer functions use a hybrid approach of performing a moderate number of EM iterations to refine the initial estimates of the variance parameters, and then switch to Newton Raphson iterations. By default, 25 EM iterations are performed in the lme function before switching to Newton Raphson iterations, and 15 EM iterations in the lmer function. The lme function allows for a wide variety of variance models including scaled identity, compound symmetry, autoregressive and unstructured variance matrices. Pinheiro and Bates (2000) provide an overview of fitting linear mixed models using the lme function in R and S-PLUS. The approach taken in the lmer function doesn’t allow for complex structures of the variance matrix of the random effects, but rather the variance models are kept simple and the complexity is moved, when needed, to the structure of the model matrices for the linear predictor (Bates, 2007a).

The DF algorithm of Graser et al. (1987) was implemented in the DFREML package of Meyer (1989) and Meyer (1991). More recently, the suite of programs in DFREML have been replaced by WOMBAT (Meyer, 2006b) which now incorporates the AI, EM and PXEM algorithms. If the number of variance parameters is small, that is less than 18, WOMBAT defaults to the AI algorithm, otherwise it begins with 3 iterations of the PXEM algorithm before switching to the AI algorithm. WOMBAT has been developed with the analysis of data from animal breeding programs in mind, and accommodates most models commonly fitted for such data, including standard univariate and multivariate analyses, as well as random regression analyses (Meyer, 2007). Multiple random effects with a choice of covariance structure can be fitted; the random effects can be distributed proportionally to an identity matrix or a relationship matrix among individuals, or user-defined covariance structures can be fitted for which the corresponding inverse can be supplied. WOMBAT offers a choice between full and reduced rank estimation of covariance matrices.
ASReml (Gilmour et al., 2006) is a statistical package designed to fit linear mixed models using REML. The computational efficiency of ASReml arises from using the Average Information (AI) algorithm and sparse matrix methods. This enables ASReml to efficiently analyse large and complex data sets. The core of ASReml forms the basis of the REML procedure in GenStat. ASReml-R is an R interface to the ASReml routines (Butler et al., 2007). ASReml was first developed to analyse data from a crop variety evaluation program in southern New South Wales. It has since become internationally popular, predominantly within agricultural research for the analysis of plant and animal breeding data. These data sets can be reasonably large and it is often desirable to fit linear mixed models with complex variance structures. ASReml allows a very wide range of variance structures to be fitted including scaled identity, compound symmetry, autoregressive and unstructured variance matrices as well as factor analytic variance models. Some examples of analyses involving the use of ASReml and the AI algorithm can be found in Gilmour et al. (1997), Cullis et al. (1998), Verbyla et al. (1999), Cullis et al. (2000) and Smith et al. (2001a).

The REML directive in GenStat accesses ASReml routines for REML estimation of variance parameters in linear mixed models. The METHOD option specifies whether to use the AI (Average Information) algorithm with sparse matrix methods to maximize the residual likelihood, or Fisher scoring with full matrix manipulation. By default, the sparse Average Information algorithm is used. The AI algorithm generally runs faster per iteration than Fisher scoring and uses much less workspace, but it may require slightly more iterations to reach convergence. Galwey (2006) provides an overview of fitting linear mixed models in GenStat.

1.3 Improved Iterative Schemes

The work in this thesis has been motivated by problems in reliability of convergence encountered when fitting linear mixed models in ASReml using the AI algorithm for REML estimation of variance parameters. As previously discussed, the AI algorithm is sensitive to the choice of starting values; failing to converge if the starting values are too far from the REML solution. The EM and PXEM algorithms are good alternatives in these situations. However they can be very slow to converge. The aim of this research is to develop an optimal iterative strategy, that is robust to choice of starting values while achieving rapid convergence, for a wide class of models.

We present a series of hybrid schemes that use EM-type iterations initially to obtain variance parameter estimates that are in the neighbourhood of the REML solution, and then switch to AI iterations to ensure rapid convergence. We begin with EM/AI and PXEM/AI schemes then present a computationally ”cheaper” alternative to using full EM or PXEM iterations in the hybrid schemes that use an internal or so-called local EM or PXEM scheme. The convergence problems with the AI algorithm are often related to the parameters in the random effects variance matrix, rather than the error variance matrix (as defined in Chapter 2). These local schemes update only the random effect variance parameters, with the estimates of the residual error variance parameters and the fixed effects held fixed at the values from the external loop.
The `lmer` function in R uses 15 EM iterations and then switches to NR iterations. If the number of variance parameters is greater than 18, `WOMBAT` uses 3 PXEM iterations then switches to AI iterations. `ASReml` has an option to use an EM iteration when an AI iteration results in variance parameter estimates outside of the parameter space. Rather than using a fixed number of EM-type iterations, or only using an EM-type iteration when an AI iteration fails, we investigate techniques for detecting when the AI algorithm is likely to fail to converge, and hence invoking EM-type iterations before an AI iteration fails. We also assess when the EM updates are close enough to the REML solution to safely switch back to AI iterations to ensure the scheme converges quickly. We also look at methods for obtaining starting values for the iterative schemes and compare the performance of the hybrid schemes to the performance of the AI algorithm with these informed starting values.

The convergence problems encountered when using the AI algorithm are often related to the parameters in the random effects variance matrix, rather than the error variance matrix (Cullis et al., 2004). And so in this thesis we focus on models with a simple structure for the error variance matrix, that is we assume that the errors and independent and identically distributed, and we look at a range of random effects variance models. We investigate the performance of the various schemes for a simple variance components model and a more complex model with an unstructured random effects variance matrix to determine an optimal iterative strategy that will work for a range of different random effects variance models. A number of data sets are used, including published data sets and simulated data. We compare the performance of the basic algorithms to that of the various composite algorithms, using both uninformed and informed starting values.

### 1.4 Outline

We begin in Chapter 2 by establishing a general framework for linear mixed models including estimation of fixed and random effects and variance parameters. Chapter 3 describes the NR, FS and AI algorithms along with a computational method for implementing the AI algorithm. Chapter 4 presents a general formulation and then the linear mixed model formulation of the EM algorithm. The PXEM algorithm is presented in Chapter 5. Chapter 6 then looks at the theoretical convergence rates for the AI, EM and PXEM algorithms. The hybrid schemes and methods for obtaining starting values are presented in Chapter 7.

In Chapters 8 and 9 we consider two different linear mixed models. Both models assume a simple variance structure for the error variance matrix, that is we assume that the errors are independent and identically distributed. The models differ in their random effects variance structure. In Chapter 8 we consider a simple variance components model with a single random effect, with a scaled identity matrix as the random effects variance matrix. In Chapter 9 we consider a model with a more complicated random effects variance matrix, a direct product of an unstructured variance matrix and the identity matrix, that can be used for random coefficient regression and the analysis of data from multi-environment plant variety trials and in multi-trait animal breeding applications. We present the form of the updating equations for the variance parameters for the AI, EM and PXEM algorithms.
and the local EM and local PXEM schemes. The performance of the various iterative schemes is compared using a number of published and simulated data sets. Firstly, we compare the performance of the AI, EM and PXEM algorithms, including the theoretical and empirical convergence rates. For the data sets for which the AI algorithm does not converge, we investigate the use of the hybrid schemes, including selection of the number of internal local iterations for the local schemes and the cut-offs for the update and score criteria for the hybrid schemes. We complete both of these chapters with the development of an optimal iterative strategy. Chapter 10 presents a summary of the thesis and our recommended iterative strategy that combines the robustness of EM-type iterations to the choice of starting values and the rapid convergence of the AI algorithm.
Chapter 2

Mixed Models

2.1 Model

Much statistical modelling involves linear models. In an ordinary linear model, all effects apart from the error term are regarded as fixed. The linear model is given by

\[ y = X\tau + e \]

where \( y \) is vector of observations, \( X \) is a matrix of known covariates, \( \tau \) is a vector of unknown fixed regression coefficients and \( e \) is a vector of random errors. In a linear mixed model, some of the coefficients are considered to be realisations of random variables rather than fixed. The inclusion of random effects allows us to model correlations between observations within the same group.

The linear mixed model is given by

\[ y = X\tau + Zu + e \]  \hspace{1cm} (2.1)

where \( y \) is a \( n \times 1 \) vector of observations, \( \tau \) is a \( t \times 1 \) vector of fixed effects, \( X \) is the associated \( n \times t \) design matrix of (assumed for convenience) full column rank, \( u \) is a \( b \times 1 \) vector of random effects, \( Z \) is the associated \( n \times b \) design matrix, and \( e \) is a \( n \times 1 \) vector of residuals. It is assumed that the joint distribution of \( u \) and \( e \) is given by

\[ \begin{bmatrix} u \\ e \end{bmatrix} \sim N \left( \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \theta \begin{bmatrix} G & 0 \\ 0 & R \end{bmatrix} \right) \]

where \( G = G(\gamma) \), and \( R = \sigma^2\Sigma \) where \( \Sigma = \Sigma(\phi) \). The vectors \( \gamma \) and \( \phi \) are vectors of variance parameters associated with \( u \) and \( e \) respectively. The parameters \( \sigma^2 \) and \( \theta \) are variance parameters which we will refer to as the residual variance scale parameter and the overall scale parameter respectively. Thus the variance parameters are \( \theta \) and \( \kappa = (\gamma', \sigma^2, \phi')' \). \( G \), \( R \) and \( \Sigma \) are variance matrices and hence are assumed to be symmetric positive definite. The distribution of \( y \) is given by

\[ y \sim N(X\tau, \theta H) \]

where \( H = ZGZ' + R \) is assumed to be symmetric positive definite.
The variance parameters $\theta$ and $\sigma^2$ are included to allow for different parameterisations. If we include both $\theta$ and $\sigma^2$ we find that the two parameters are unidentifiable as $\text{Var}(u) = \theta G(\gamma)$ and $\text{Var}(e) = \theta \sigma^2 \Sigma(\phi)$. In practice at least one of these parameters is set to one.

If $\theta = 1$ and $\sigma^2 = 1$ then $\text{Var}(u) = G(\gamma)$ and $\text{Var}(e) = \Sigma(\phi)$ and so the elements of $\gamma$ and $\phi$ are components of ”variance”. The variance parameters to be estimated are $\kappa = (\gamma', \phi')'$.

If $\theta = 1$ and $\sigma^2 \neq 1$ then $\text{Var}(u) = G(\gamma)$ and $\text{Var}(e) = \sigma^2 \Sigma(\phi)$ and so the elements of $\gamma$ are components and the elements of $\phi$ are ratios relative to $\sigma^2$. The variance parameters to be estimated are $\kappa = (\gamma', \sigma^2, \phi')'$.

If $\theta \neq 1$ and $\sigma^2 = 1$ then $\text{Var}(u) = \theta G(\gamma)$ and $\text{Var}(e) = \theta \Sigma(\phi)$ and so the elements of $\gamma$ and $\phi$ are ratios relative to $\theta$. The variance parameters to be estimated are $\theta$ and $\kappa = (\gamma', \phi')'$.

Note that in this thesis we only consider the performance of iterative schemes for models in which $\theta = 1$. However, we develop the theory in terms of $\theta$ and $\sigma^2$ for the sake of generality.

The model in (2.1) can include more than one random term, that is

$$y = X\tau + Z_1 u_1 + Z_2 u_2 + \ldots + Z_q u_q + e = X\tau + Zu + e$$

In this case the vector of random effects $u$ is composed of $q$ subvectors such that $u = (u'_1, u'_2, \ldots, u'_q)'$ where the subvector $u_i$ is the vector of effects for the $i$th random term and is of length $b_i \times 1$. Each subvector $u_i$ has an associated design matrix $Z_i$ of size $n \times b_i$ such that $Z$ is of the form $Z = [Z_1 \ Z_2 \ \ldots \ Z_q ]$. Typically these subvectors are assumed to be independently normally distributed with variance matrices $G_i$. Thus we have

$$G = \oplus_{i=1}^q G_i = \begin{bmatrix} G_1 & 0 & \ldots & 0 & 0 \\ 0 & G_2 & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & G_{q-1} & 0 \\ 0 & 0 & \ldots & 0 & G_q \end{bmatrix}$$

(2.2)

The variance parameter $\gamma$ is also partitioned conformably with the partitioning of $G$ such that $G_i = G_i(\gamma_i)$, that is $\gamma = (\gamma_1', \gamma_2', \ldots, \gamma_q')'$. The submatrices $G_i$ may take many forms.

Note that in general in this thesis we consider models that have a single random effect $u$ that is not made up of subvectors. However the theory will be developed to allow for models with the above variance structure for the sake of generality.

In most cases the vector of residuals $e$ represents the random errors for a single set of data. However in some situations the vector $e$ will be a series of vectors indexed by a factor or factors. Thus $e = (e'_1, e'_2, \ldots, e'_s)'$ where $e_j$ represent the errors of sections of the data. These sections may represent different populations, trials or experiments in a
2.2. ESTIMATION

series. It is assumed that

$$\Sigma = \phi_j = 1 \sum_{j=1}^s \Sigma_j = \begin{bmatrix}
\Sigma_1 & 0 & \ldots & 0 & 0 \\
0 & \Sigma_2 & \ldots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \ldots & \Sigma_{s-1} & 0 \\
0 & 0 & \ldots & 0 & \Sigma_s
\end{bmatrix}$$

so that each section has its own variance matrix but they are assumed independent. The variance parameter $\phi$ is partitioned so that $\Sigma_i = \Sigma_i(\phi_i)$, that is $\phi = (\phi'_1, \phi'_2, \ldots, \phi'_s)'$.

Note that in general in this thesis we consider models for single data sets only and therefore the theory will be developed for a non-partitioned vector of residuals.

2.2 Estimation

Estimation in a linear mixed model involves two linked processes. For a given sample $y$ and given design matrices $X$ and $Z$, we wish to estimate the fixed and random effects, $\tau$ and $u$, and the variance parameters, $\theta$ and $\kappa$. First we will consider estimation of the fixed and random effects for given $\theta$ and $\kappa$. This is done using the method first established by Henderson (1950) and Henderson (1963) which later became known as best linear unbiased prediction. We will then consider estimation of the variance parameters for given $\tau$ and $u$. The preferred method of estimation of variance parameters in linear mixed models is residual maximum likelihood (REML) estimation (Patterson and Thompson (1971)).

2.2.1 Best Linear Unbiased Prediction

Conventionally estimators of fixed effects are called estimators, whereas estimators of random effects are called predictors. Hence we want to estimate fixed effects $\tau$ by $\hat{\tau}$ and predict random effects $u$ by $\tilde{u}$. Best linear unbiased prediction results in $\hat{\tau}$, the best linear unbiased estimator (BLUE), and in $\tilde{u}$, the best linear unbiased predictor (BLUP). Robinson (1991) explains that the BLUPs of the random effects are linear in the sense that they are linear functions in the data $y$; unbiased in the sense that the expected value of the estimate is equal to the true value of the quantity being estimated; best in the sense that the mean square error is minimised within the class of linear unbiased estimators; and predictors to distinguish then from estimators of fixed effects. Robinson (1991) presents a number of derivations of BLUP. One is the derivation described by Henderson (1950) and Henderson (1963), another is a Bayesian derivation as given by Lindley and Smith (1972) and Dempfle (1977). The derivations of Goldberger (1962) and Harville (1976) are also presented in Robinson’s paper.

Here we will present the derivation of Henderson (1963). Let $a'y$ be a linear unbiased estimator of $c'_1 \tau + c'_2 u$ where $c_1$ and $c_2$ are $t \times 1$ and $b \times 1$ vectors respectively so that

$$E[a'y] = E[c'_1 \tau + c'_2 u]$$
Now,
\[
E[a'y] = a'E[y] = a'E[X\tau + Zu + e] = a'X\tau
\]
and
\[
E[c'_1\tau + c'_2u] = c'_1\tau
\]
Hence
\[
a'X\tau = c'_1\tau
\]
This results in
\[
X'a = c_1 \quad (2.3)
\]
To find the best linear unbiased predictor we need to minimise the mean square error (MSE) given by
\[
\text{MSE} = E[(a'y - c'_1\tau - c'_2u)^2] = E[[a'y - c'_2u]^2] - 2E[(a'y - c'_2u)]E[c'_1\tau] + E[(c'_1\tau)^2]
\]
\[
= \text{Var}[a'y - c'_2u] + (E[a'y - c'_2u])^2 - 2E[(a'y - c'_2u)]E[c'_1\tau] + E[(c'_1\tau)^2]
\]
\[
= \text{Var}[a'y - c'_2u] + (a'X\tau)^2 - 2a'X\tau c'_1\tau + (c'_1\tau)^2
\]
\[
= \text{Var}[a'y - c'_2u] + (a'X\tau - c'_1\tau)^2
\]
\[
= \text{Var}[a'y - c'_2u] \quad \text{(using 2.3)}
\]
\[
= a'\text{Var}[y]a + c'_2\text{Var}[u]c_2 - 2a'\text{Cov}[y, u]c_2
\]
\[
= a'\theta Ha + c'_2Gc_2 - 2a'ZGc_2
\]
\[
= \theta(a'Ha + c'_2Gc_2 - 2a'ZGc_2)
\]
This can be minimised using Lagrangian Multipliers subject to the constraint given in (2.3). The function to be minimised is
\[
M = \text{MSE} + 2\lambda'(c_1 - X'a) = \theta(a'Ha + c'_2Gc_2 - 2a'ZGc_2) + 2\lambda'(c_1 - X'a)
\]
where \(\lambda\) is the vector of Lagrangian multipliers. To minimise M we take partial derivatives with respect to \(a\) and \(\lambda\), and equate to zero. Using Result A.1 the partial derivative of M with respect to \(a\) is
\[
\frac{\partial M}{\partial a} = 2\theta Ha - 2\theta ZGc_2 - 2X\lambda
\]
and equating to zero and solving for \(a\) gives
\[
a = H^{-1}ZGc_2 + (\theta)^{-1}H^{-1}X\lambda \quad (2.4)
\]
The partial derivative of M with respect to \(\lambda\) is
\[
\frac{\partial M}{\partial \lambda} = 2(c_1 - X'a)
and equating to zero and substituting in \(a\) as given in 2.4, gives

\[
c_1 = X'a \\
= X'H^{-1}ZGc_2 + \theta^{-1}X'H^{-1}X\lambda
\]

Rearranging to solve for \(\lambda\) gives

\[
\lambda = \theta(X'H^{-1}X)^{-1}(c_1 - X'H^{-1}ZGc_2)
\]

Substituting into (2.4) allows \(a\) to be written as

\[
a = H^{-1}ZGc_2 + \theta^{-1}H^{-1}\theta(X'H^{-1}X)^{-1}(c_1 - X'H^{-1}ZGc_2) \\
= H^{-1}ZGc_2 + H^{-1}X(X'H^{-1}X)^{-1}c_1 - H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1}ZGc_2 \\
= (H^{-1} - H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1})ZGc_2 + H^{-1}X(X'H^{-1}X)^{-1}c_1 \\
= PZGc_2 + H^{-1}X(X'H^{-1}X)^{-1}c_1
\]

where

\[
P = H^{-1} - H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1} \quad (2.5)
\]

Hence \(a'y\) is given by

\[
a'y = c_1'(X'H^{-1}X)^{-1}X'H^{-1}y + c_2'GZ'Py \\
= c_1'\hat{\tau} + c_2'\tilde{u} \quad \text{say.}
\]

Thus the best linear unbiased predictor \(a'y\) of \(c_1'(X'H^{-1}X)^{-1}X'H^{-1}y\) is given by \(c_1'\hat{\tau} + c_2'\tilde{u}\) where

\[
\hat{\tau} = (X'H^{-1}X)^{-1}XH^{-1}y \quad (2.6) \\
\tilde{u} = GZ'Py \quad (2.7)
\]

Equation (2.6) is the BLUE of \(\tau\) and equation (2.7) is the BLUP of \(u\).

Henderson (1950) proposed an alternative method of obtaining these solutions; estimation of \(\tau\) and prediction of \(u\), by maximising the so-called joint likelihood of \((y, u)\). Note that this is not strictly a joint likelihood as \(u\) is not observed. The joint density of \(y\) and \(u\) can be written as the product of the conditional density of \(y\) given \(u\) and the density of \(u\),

\[
f(y, u; \tau, \theta, \kappa) = f(y|u; \tau, \theta, \phi)f(u; \theta, \gamma)
\]

where

\[
y|u \sim N(X\tau + Zu, \theta R) \\
u \sim N(0, \theta G)
\]

The log-likelihood of \(y\) given \(u\), excluding constant terms, is given by

\[
\ell(\tau, \theta, \phi; y|u) = -\frac{1}{2}\left(n \log \theta + \log |R| + \frac{(y - X\tau - Zu)'R^{-1}(y - X\tau - Zu)}{\theta}\right)
\]

Note that this is not strictly a log-likelihood as \(u\) is not observed. Technically it is really the log-density function of \(y|u\) but for our purposes we will call it a log-likelihood.
The log-likelihood of \( u \), excluding constant terms, is given by

\[
\ell(\theta, \gamma; u) = -\frac{1}{2} \left( q \log \theta + \log |G| + \frac{u'G^{-1}u}{\theta} \right)
\]

We can now write the joint log-likelihood of \( y \) and \( u \), excluding constant terms, as

\[
\ell(\tau, \theta, \kappa; y, u) = \ell(\tau, \theta, \phi; y|u) + \ell(\theta, \gamma; u) = -\frac{1}{2} \left( n \log \theta + \log |R| + \frac{(y - X\tau - Zu)'R^{-1}(y - X\tau - Zu)}{\theta} \right) - \frac{1}{2} \left( q \log \theta + \log |G| + \frac{u'G^{-1}u}{\theta} \right)
\]

Note again that this is not strictly a log-likelihood.

Estimates of \( \tau \) and \( u \) are found by maximising the joint log-likelihood in (2.8). Using Result A.1 the derivative of (2.8) with respect to \( \tau \) is

\[
\frac{\partial \ell(\tau, \theta, \kappa; y, u)}{\partial \tau} = X'R^{-1}(y - X\tau - Zu) \theta
\]

and equating to zero gives

\[
X'R^{-1}X\hat{\tau} + X'R^{-1}Z\hat{u} = X'R^{-1}y \tag{2.9}
\]

The derivative of (2.8) with respect to \( u \) is given by

\[
\frac{\partial \ell(\tau, \theta, \kappa; y, u)}{\partial u} = Z'R^{-1}(y - X\tau - Zu) \theta - G^{-1}u \theta
\]

and equating to zero gives

\[
Z'R^{-1}X\hat{\tau} + (Z'R^{-1}Z + G^{-1})\hat{u} = Z'R^{-1}y \tag{2.10}
\]

Equations (2.9) and (2.10) are known as the mixed model equations and are most commonly represented in matrix form as given below

\[
\begin{bmatrix}
X'R^{-1}X & X'R^{-1}Z \\
Z'R^{-1}X & Z'R^{-1}Z + G^{-1}
\end{bmatrix}
\begin{bmatrix}
\hat{\tau} \\
\hat{u}
\end{bmatrix}
= 
\begin{bmatrix}
X'R^{-1}y \\
Z'R^{-1}y
\end{bmatrix}
\tag{2.11}
\]

The solutions in (2.6) and (2.7), can be calculated as solutions to the mixed model equations as follows. Let \( C \) be the coefficient matrix on the left hand side of (2.11) that is

\[
C = 
\begin{bmatrix}
X'R^{-1}X & X'R^{-1}Z \\
Z'R^{-1}X & Z'R^{-1}Z + G^{-1}
\end{bmatrix}
\tag{2.12}
\]

and denote the components by

\[
C = 
\begin{bmatrix}
C_{XX} & C_{XZ} \\
C_{ZX} & C_{ZZ}
\end{bmatrix}
\]

The mixed model equations in (2.9) and (2.10) can now be written as

\[
C_{XX}\hat{\tau} + C_{XZ}\hat{u} = b_X \tag{2.13}
\]

\[
C_{ZX}\hat{\tau} + C_{ZZ}\hat{u} = b_Z \tag{2.14}
\]
where \( b_X = X'R^{-1}y \) and \( b_Z = Z'R^{-1}y \). Writing the second set of equations (2.14) in terms of \( \tilde{u} \) gives

\[
\tilde{u} = C_{ZZ}^{-1}(b_Z - C_{ZX}\hat{\tau})
\]

and substituting this into the first set of equations (2.13) gives

\[
C_{XX}\hat{\tau} + C_{XZ}C_{ZZ}^{-1}(b_Z - C_{ZX}\hat{\tau}) = b_X
\]

Writing this equation in terms of \( \hat{\tau} \) gives

\[
\hat{\tau} = (C_{XX} - C_{XZ}C_{ZZ}^{-1}C_{XX})^{-1}(b_X - C_{XZ}C_{ZZ}^{-1}b_Z)
\]

Now,

\[
C_{XX} - C_{XZ}C_{ZZ}^{-1}C_{XX} = X'R^{-1}X - X'R^{-1}Z(Z'R^{-1}Z + G^{-1})^{-1}Z'R^{-1}X
\]

\[
= X'(R^{-1} - R^{-1}Z(Z'R^{-1}Z + G^{-1})^{-1}Z'R^{-1})X
\]

\[
= X'(R + ZGZ')^{-1}X
\]

(using Result A.3)

Thus

\[
\hat{\tau} = (X'H^{-1}X)^{-1}(X'R^{-1}y - X'R^{-1}Z(Z'R^{-1}Z + G^{-1})^{-1}Z'R^{-1}y)
\]

\[
= (X'H^{-1}X)^{-1}X'(R^{-1} - R^{-1}Z(Z'R^{-1}Z + G^{-1})^{-1}Z'R^{-1})y
\]

\[
= (X'H^{-1}X)^{-1}X'H^{-1}y
\]

This is the BLUE of \( \tau \) as given in (2.6).

The solution for \( \tilde{u} \) is found by back-substituting \( \hat{\tau} \) into the second set of equations to give

\[
\tilde{u} = (Z'R^{-1}Z + G^{-1})^{-1}(Z'R^{-1}y - Z'R^{-1}X(X'H^{-1}X)^{-1}X'H^{-1}y)
\]

\[
= (Z'R^{-1}Z + G^{-1})^{-1}Z'R^{-1}(I - X(X'H^{-1}X)^{-1}X'H^{-1})y
\]

\[
= GZ'(R + ZGZ')^{-1}(I - X(X'H^{-1}X)^{-1}X'H^{-1})y
\]

(using Result A.3)

\[
= GZ'H^{-1}(I - X(X'H^{-1}X)^{-1}X'H^{-1})y
\]

\[
= GZ'Py
\]

\[
= GZ'H^{-1}(y - X\hat{\tau})
\]

This is the BLUP of \( u \) as given in (2.7). Hence the solutions to the mixed model equations, \( \hat{\tau} \) and \( \tilde{u} \), are the BLUE of \( \tau \) and the BLUP of \( u \) as shown in (2.6) and (2.7).

We can also obtain estimates of the residuals

\[
\hat{e} = y - X\hat{\tau} - \tilde{u}
\]

\[
= y - X\hat{\tau} - ZGZ'H^{-1}(y - X\hat{\tau})
\]

\[
= y - X\hat{\tau} - ZGZ'H^{-1}y + ZGZ'H^{-1}X\hat{\tau}
\]

\[
= (I - ZGZ'H^{-1})y - (I - ZGZ'H^{-1})X\hat{\tau}
\]

\[
= (I - ZGZ'H^{-1})y - (I - ZGZ'H^{-1})X(X'H^{-1}X)^{-1}X'H^{-1}y
\]

\[
= (I - ZGZ'H^{-1})(I - X(X'H^{-1}X)^{-1}X'H^{-1})y
\]

\[
= (H - ZGZ')(H^{-1} - H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1})y
\]

\[
= RP'y
\]
A more computationally convenient way of obtaining $\hat{\tau}$ and $\tilde{e}$ is to write the mixed model equations in (2.11) as

$$C\tilde{\beta} = W'R^{-1}y$$

where $C$ is the left hand coefficient matrix of the mixed model equations as defined in (2.12), $\tilde{\beta} = \begin{bmatrix} \hat{\tau} \\ \tilde{u} \end{bmatrix}$ and $W = \begin{bmatrix} X & Z \end{bmatrix}$. An estimate of $\beta$ is given by

$$\tilde{\beta} = C^{-1}W'R^{-1}y$$

To obtain this estimate, $C^{-1}$ is required.

The coefficient matrix of the mixed model equations, $C$, as given in (2.12) is

$$C = \begin{bmatrix} C_{XX} & C_{XZ} \\ C_{ZX} & C_{ZZ} \end{bmatrix} = \begin{bmatrix} X'R^{-1}X & X'R^{-1}Z \\ Z'R^{-1}X & Z'R^{-1}Z + G^{-1} \end{bmatrix}$$

Using Result A.12, the inverse of $C$ is given by

$$C^{-1} = \begin{bmatrix} C_{XX} & X^{XZ} \\ C_{ZX} & C_{ZZ} \end{bmatrix} = \begin{bmatrix} C_{XX} & X^{XZ} \\ -C_{ZZ}^{-1}C_{ZX}C_{XX} & C_{XX}^{-1} - C_{XX}C_{ZZ}^{-1}C_{ZX}C_{XX}^{-1}C_{ZZ}^{-1} \end{bmatrix}$$

(2.17)

where $C_{XX} = (C_{XX} - C_{XZ}C_{ZZ}^{-1}C_{ZX})^{-1}$.

We know from (2.15) that

$$C_{XX} - C_{XZ}C_{ZZ}^{-1}C_{ZX} = X'H^{-1}X$$

Hence $C_{XX}$ is given by

$$C_{XX} = (X'H^{-1}X)^{-1}$$

Using Result A.3 we find that

$$(Z'R^{-1}Z + G^{-1})^{-1}Z'R^{-1} = GZ'(R + ZGZ')^{-1} = GZ'H^{-1}$$
2.2. ESTIMATION

and as a result the other elements of $C^{-1}$ are given by

\[
C^{xz} = -C^{xx}C_{xz}C_{zz}^{-1} \\
= -(X'H^{-1}X)^{-1}X'R^{-1}Z(Z'R^{-1}Z + G^{-1})^{-1} \\
= -(X'H^{-1}X)^{-1}X'H^{-1}ZG
\]

\[
C^{zx} = -C_{zz}^{-1}C_{xz}C^{xx} \\
= -(Z'R^{-1}Z + G^{-1})^{-1}Z'R^{-1}X(X'H^{-1}X)^{-1}X'R^{-1}Z(Z'R^{-1}Z + G^{-1})^{-1} \\
= -(Z'R^{-1}Z + G^{-1})^{-1} + GZ'H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1}ZG \\
= (Z'R^{-1}Z + G^{-1})^{-1} + GZ(H^{-1} - P)ZG \\
= (Z'R^{-1}Z + G^{-1})^{-1} + GZ'H^{-1}XG - GZ'PZG \\
= (Z'R^{-1}Z + G^{-1})^{-1}(I + Z'R^{-1}ZG) - GZ'PZG \\
= (Z'R^{-1}Z + G^{-1})^{-1}(Z'R^{-1}Z + G^{-1})G - GZ'PZG \\
= G - GZ'PZG \tag{2.19}
\]

\[
C^{zz} = C_{zz}^{-1} + C_{zz}^{-1}C_{xz}C_{xx}C_{xz}C_{zz}^{-1} \\
= (Z'R^{-1}Z + G^{-1})^{-1} + GZ'H^{-1}XG - GZ'PZG \\
= (Z'R^{-1}Z + G^{-1})^{-1}(I + Z'R^{-1}ZG) - GZ'PZG \\
= (Z'R^{-1}Z + G^{-1})^{-1}(Z'R^{-1}Z + G^{-1})G - GZ'PZG \\
= G - GZ'PZG \tag{2.20}
\]

Therefore

\[
C^{-1} = \begin{bmatrix}
C^{xx} & C^{xz} \\
C^{zx} & C^{zz}
\end{bmatrix} \\
= \begin{bmatrix}
(X'H^{-1}X)^{-1} & -(X'H^{-1}X)^{-1}X'H^{-1}ZG \\
-GZ'H^{-1}X(X'H^{-1}X)^{-1} & G - GZ'PZG
\end{bmatrix} \tag{2.21}
\]

Note that to calculate estimates of the fixed and random effects we need values for the variance matrices $G$ and $R$, hence we require the variance parameters $\kappa$. In practice, these are unknown and must be replaced by estimates from the data. When the REML estimates for $\kappa$ are substituted into the BLUE in (2.6) and the BLUP in(2.7), the resulting estimates are referred to as the Empirical BLUE (E-BLUE) and the Empirical BLUP (E-BLUP).

2.2.2 REML Estimation of Variance Parameters

Hartley and Rao (1967) developed a methodology for the estimation of variance parameters for a wide range of models using maximum likelihood (ML) estimation. Estimates for $\tau$, $\theta$ and $\kappa$ are obtained by maximising the log-likelihood for $y$. The log-likelihood for $y$, omitting any constant terms, is given by

\[
\ell(\tau, \theta, \kappa; y) = -\frac{1}{2} \left( n \log \theta + \log |H| + \frac{(y - X\tau)'H^{-1}(y - X\tau)}{\theta} \right)
\]
ML estimation of variance parameters tends to result in biased estimates. This fact led to the development of restricted or residual maximum likelihood (REML) estimation. Patterson and Thompson (1971) modified the ML procedure to account for the loss of degrees of freedom due to fitting the fixed effects. REML estimation does not maximise the full likelihood of \( y \), but rather the likelihood of a set of errors contrasts, that is the likelihood based on the residuals obtained after fitting the fixed effects, hence taking into account the loss of degrees of freedom associated with the estimation of fixed effects (Searle et al. (1992)). Note that for balanced data, the REML estimates of the variance parameters for linear mixed models are equal to the ANOVA estimates. REML estimates of the variance parameters are less biased than the ML estimates, hence making REML estimation the preferred method.

The derivation of the likelihood as presented by Patterson and Thompson (1971) is complicated. Harville (1974) presents a Bayesian derivation of the likelihood function, while Harville (1977) presents a review of the maximum likelihood approach to variance parameter estimation including an alternative derivation of the REML likelihood. Another derivation of the likelihood is presented in Cooper and Thompson (1977).

Verbyla (1990) presents an enlightening derivation of the REML function related to the derivation of Harville (1974). The full log-likelihood of \( y \) as used in ML estimation, is partitioned into a conditional log-likelihood and a marginal log-likelihood. Maximisation of the conditional log-likelihood results in estimates of the fixed effects which are the same as those given by ML estimation. Maximisation of the marginal log-likelihood provides estimates of the variance parameters \( \gamma \) and \( \phi \). The marginal log-likelihood is known as the residual or REML log-likelihood.

Verbyla (1990) considers a non-singular matrix \( L = [L_1, L_2] \) where \( L_1 \) and \( L_2 \) are \( n \times t \) and \( n \times (n-t) \) matrices respectively, that are chosen to satisfy \( L_1'X = I_t \) and \( L_2'X = 0 \). The data is transformed from \( y \) to \( L'y \) where

\[
L'y = \begin{bmatrix} L_1'y \\ L_2'y \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}
\]

and this transformed data has a distribution given by

\[
\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \sim N \left( \begin{bmatrix} \tau \\ 0 \end{bmatrix}, \theta \begin{bmatrix} L_1'HL_1 & L_1'HL_2 \\ L_2'HL_1 & L_2'HL_2 \end{bmatrix} \right)
\]

The density function of \( L'y \) can be written as the product of the conditional density of \( y_1 \) given \( y_2 \) and the marginal density of \( y_2 \). The marginal distribution of \( y_2 \) is

\[
y_2 \sim N(0, \theta L_2'HL_2)
\]

and the marginal log-likelihood of \( y_2 \), using Result A.4 and omitting constant terms, is given by

\[
\ell(\theta, \kappa; y_2) = -\frac{1}{2} \left( (n-t) \log \theta + \log |L_2'HL_2| + \frac{y_2'(L_2'HL_2)^{-1}y_2}{\theta} \right) = -\frac{1}{2} \left( (n-t) \log \theta + \log |L_2'HL_2| + \frac{y'y(L_2'HL_2)^{-1}L_2'y}{\theta} \right) = -\frac{1}{2} \left( (n-t) \log \theta + \log |L_2'HL_2| + \frac{y'Py}{\theta} \right)
\]
The marginal distribution of $y_1$ given $y_2$ is

$$y_1|y_2 \sim N \left( \tau + L_1'HL_2(L_2'HLL_2)^{-1}y_2, \theta(L_1'HLL_1 - L_1'HLL_2(L_2'HLL_2)^{-1}L_2'HLL_1) \right)$$

Using Result A.4 the conditional variance matrix is

$$L_1'HL_1 - L_1'HLL_2(L_2'HLL_2)^{-1}L_2'HLL_1$$

$$= L_1'(H - HLL_2(L_2'HLL_2)^{-1}L_2'H)L_1$$

$$= L_1'X(X'H^{-1}X)^{-1}X'L_1$$

$$= (X'H^{-1}X)^{-1}$$

since $L_1'X = I_t$. Hence the conditional distribution of $y_1$ given $y_2$ can be written as

$$y_1|y_2 \sim N(\tau + y_2^*, \theta(X'H^{-1}X)^{-1})$$

where $y_2^* = L_1'HL_2(L_2'HLL_2)^{-1}y_2$. This allows us to write the conditional log-likelihood of $y_1$ given $y_2$, omitting constant terms, as

$$\ell(\tau, \theta, \kappa; y_1|y_2) = \frac{1}{2} \left( -\log \theta + \log |(X'H^{-1}X)^{-1}| + \frac{(y_1 - \tau - y_2^*)'(X'H^{-1}X)(y_1 - \tau - y_2^*)}{\theta} \right)$$

(2.24)

The marginal log-likelihood for $y_2$ contains no information on $\tau$ and so the conditional log-likelihood of $y_1$ given $y_2$ must be used. So for given $\kappa$ we differentiate (2.24) with respect to $\tau$ using Result A.1 to get

$$\frac{\partial \ell(\tau, \theta, \kappa; y_1|y_2)}{\partial \tau} = X'H^{-1}X(y_1 - y_2^* - \tau)$$

and equate to zero to get

$$\hat{\tau} = y_1 - y_2^*$$

$$= y_1 - L_1'HL_2(L_2'HLL_2)^{-1}y_2$$

$$= L_1'y - L_1'HLL_2(L_2'HLL_2)^{-1}L_2'y$$

$$= L_1'(H - HLL_2(L_2'HLL_2)^{-1}L_2'H)H^{-1}y$$

$$= L_1'X(XH^{-1}X)^{-1}X'H^{-1}y$$

(using Result A.4)

$$= (X'H^{-1}X)^{-1}X'H^{-1}y$$

which is the BLUE as given in (2.6).

The log-likelihood of $y_1$ given $y_2$ given in (2.24) is a function of both $\tau$ and $(\theta, \kappa)$. However since $\hat{\tau} = y_1 - y_2^*$, then $\hat{\tau}$ and $y_1$ are both of length $t$. Therefore once $\tau$ has been estimated then there is no information left to estimate $H$ and hence the variance parameters $(\theta, \kappa)$.

To estimate $\kappa$ we need to look at the marginal distribution of $y_2$. The marginal log-likelihood of $y_2$ as given in (2.23) is

$$\ell_R(\theta, \kappa; y_2) = -\frac{1}{2} \left( (n - t) \log \theta + \log |L_2'HL_2| + \frac{y'Py}{\theta} \right)$$

(2.25)
This is known as the REML log-likelihood.

The log-likelihood of $L'y$ can be written as the sum of the conditional log-likelihood of $y_1$ given $y_2$ and the marginal log-likelihood of $y_2$, that is

$$\ell(\tau, \kappa; L'y) = \ell(\tau, \kappa; y_1 | y_2) + \ell(\kappa; y_2)$$

This allows the log-determinants to be equated as follows

$$\log |L'HL| = \log |(X'H^{-1}X)^{-1}| + \log |L'_2H_L2|$$

and hence

$$\log |L'_2H_L2| = \log |L'L| + \log |H| + \log |X'H^{-1}X|$$

Therefore the REML log-likelihood in (2.25) can be expressed as (omitting terms that do not involve $\theta$ or $\kappa$)

$$\ell_R(\theta, \kappa; y_2) = -\frac{1}{2} \left( (n-t) \log \theta + \log |H| + \log |X'H^{-1}X| + \frac{y'Py}{\theta} \right)$$

(2.26)

We will now look at a more computationally convenient way of calculating the REML log-likelihood. Using Result A.2 and (2.15) we have

$$|C| = |C_{ZZ}| |C_{XX} - C_{XZ}C_{ZZ}^{-1}C_{ZX}|$$

$$= |ZR^{-1}Z' + G^{-1}|X'H^{-1}X|$$

Again, using Result A.2 we find

$$|ZR^{-1}Z' + G^{-1}| = |G^{-1}| |I + R^{-1}ZGZ'|$$

$$= |G^{-1}| |R^{-1}| |R + ZGZ'|$$

$$= |G^{-1}| |R^{-1}| |H|$$

Hence the determinant of $C$ is given by

$$|C| = |G^{-1}| |R^{-1}| |H| |X'H^{-1}X|$$

Taking the logarithm of the determinant of $C$ gives

$$\log |C| = -\log |G| - \log |R| + \log |H| + \log |X'H^{-1}X|$$

hence

$$\log |H| + \log |X'H^{-1}X| = \log |C| + \log |G| + \log |R|$$

and the REML log-likelihood in (2.26) can be written as

$$\ell_R(\theta, \kappa; y_2) = -\frac{1}{2} \left( (n-t) \log \theta + \log |G| + \log |R| + \log |C| + \frac{y'Py}{\theta} \right)$$

$$\quad = -\frac{1}{2} \left( (n-t) \log \theta + n \log \sigma^2 + \log |G| + \log |\Sigma| + \log |C| + \frac{y'Py}{\theta} \right)$$

(2.27)
The REML estimate of \( \theta \) is found by differentiating the REML log-likelihood in (2.26) with respect to \( \theta \). The REML score for \( \theta \) is given by

\[
U(\theta) = \frac{\partial \ell_R(\theta, \kappa; y_2)}{\partial \theta} = -\frac{1}{2} \left( \frac{(n-t)}{\theta} - \frac{y'Py}{\theta^2} \right) \tag{2.28}
\]

The REML estimate of \( \theta \) is obtained by equating the REML score for \( \theta \) to zero and solving for \( \theta \), that is

\[
\hat{\theta} = \frac{y'Py}{n-t} \tag{2.29}
\]

The REML estimate of \( \kappa_i \) is found by differentiating the REML log-likelihood in (2.26) with respect to \( \kappa_i \) and solving for \( \kappa_i \). The REML score for \( \kappa_i \) is given by

\[
U((\kappa); y_2) = \frac{\partial \ell_R(\kappa; y_2)}{\partial \kappa_i} = -\frac{1}{2} \left( \frac{\partial \log |H|}{\partial \kappa_i} + \frac{\partial \log |X'H^{-1}X|}{\partial \kappa_i} + \frac{1}{\theta} \frac{\partial y'Py}{\partial \kappa_i} \right) \tag{2.30}
\]

Define \( \dot{H}_i = \frac{\partial H}{\partial \kappa_i} \) and consider using Result A.1

\[
\frac{\partial \log |H|}{\partial \kappa_i} = \text{tr}(H^{-1}\dot{H}_i) \\
\frac{\partial H^{-1}}{\partial \kappa_i} = -H^{-1}\dot{H}_iH^{-1}
\]

so that

\[
\frac{\partial \log |H|}{\partial \kappa_i} + \frac{\partial \log |X'H^{-1}X|}{\partial \kappa_i} = \text{tr}(H^{-1}\dot{H}_i) - \text{tr}((X'H^{-1}X)^{-1}X'H^{-1}\dot{H}_iH^{-1}X) = \text{tr}(H^{-1}\dot{H}_i - H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1}\dot{H}_i) = \text{tr}(PH_i)
\]

and

\[
\frac{\partial y'Py}{\partial \kappa_i} = y' \frac{\partial P}{\partial \kappa_i} y = -y'PH_iPy \quad \text{(using Result A.7)}
\]

Therefore the REML score for \( \kappa_i \) in (2.30) can be written as

\[
U(\kappa_i) = -\frac{1}{2} \left( \text{tr}(PH_i) - \frac{y'P\dot{H}_iPy}{\theta} \right) \tag{2.31}
\]

To estimate \( \kappa_i \) we need to equate the REML score to zero and solve for \( \kappa_i \). In general (2.31) cannot be solved directly and an iterative scheme is required.
We will now look at explicit forms of the REML score for \( \gamma \), \( \sigma^2 \) and \( \phi \). First we need to consider the different forms of \( \mathbf{H}_i = \frac{\partial \mathbf{H}}{\partial \gamma_i} \). Recall that \( \mathbf{H} = \mathbf{ZGZ}' + \sigma^2 \mathbf{\Sigma} \).

For a random effects variance parameter \( \gamma_{ij} \) associated with \( \mathbf{G}_i \), \( \mathbf{H}_{ij} \) is given by

\[
\mathbf{H}_{ij} = \frac{\partial \mathbf{H}}{\partial \gamma_{ij}} = \frac{\partial \mathbf{ZGZ}'}{\partial \gamma_{ij}} = \frac{\partial}{\partial \gamma_{ij}} (\mathbf{Z}_1 \mathbf{G}_1 \mathbf{Z}_1' + \ldots + \mathbf{Z}_i \mathbf{G}_i \mathbf{Z}_i' + \ldots \mathbf{Z}_q \mathbf{G}_q \mathbf{Z}_q')
\]

\[
= \mathbf{Z}_i \frac{\partial \mathbf{G}_i}{\partial \gamma_{ij}} \mathbf{Z}_i'
\]

(2.32)

where \( \mathbf{G}_{ij} = \frac{\partial \mathbf{G}_i}{\partial \gamma_{ij}} \). Note that if there is only one random term in the model then

\[
\mathbf{H}_i = \frac{\partial \mathbf{H}}{\partial \gamma_i} = \mathbf{Z} \frac{\partial \mathbf{G}_i}{\partial \gamma_i} \mathbf{Z}'
\]

(2.33)

For the residual variance scale parameter \( \sigma^2 \)

\[
\mathbf{H}_i = \frac{\partial \mathbf{H}}{\partial \sigma^2} = \mathbf{\Sigma}
\]

(2.34)

and for a residual variance parameter \( \phi_i \)

\[
\mathbf{H}_i = \frac{\partial \mathbf{H}}{\partial \phi_i} = \sigma^2 \frac{\partial \mathbf{\Sigma}}{\partial \phi_i} = \sigma^2 \mathbf{\Sigma}_i
\]

(2.35)

where \( \mathbf{\Sigma}_i = \frac{\partial \mathbf{\Sigma}}{\partial \phi_i} \).

We can obtain the REML score for \( \gamma_{ij} \) by substituting \( \mathbf{H}_{ij} = \mathbf{Z}_i \mathbf{G}_{ij} \mathbf{Z}_i' \) into (2.31).

\[
U(\gamma_{ij}) = -\frac{1}{2} \left( \text{tr}(\mathbf{PZ}_i \mathbf{G}_{ij} \mathbf{Z}_i') - \frac{y' \mathbf{PZ}_i \mathbf{G}_{ij} \mathbf{Z}_i' \mathbf{Py}}{\theta} \right)
\]

(2.36)

Using Result A.5, the first term in the REML score for \( \gamma_{ij} \) in (2.36) can be written as

\[
\text{tr}(\mathbf{PZ}_i \mathbf{G}_{ij} \mathbf{Z}_i') = \text{tr}((\mathbf{R}^{-1} - \mathbf{R}^{-1} \mathbf{W} \mathbf{C}^{-1} \mathbf{W}' \mathbf{R}^{-1}) \mathbf{Z}_i \mathbf{G}_{ij} \mathbf{Z}_i')
\]

\[
= \text{tr}(\mathbf{G}_{ij} \mathbf{Z}_i'(\mathbf{R}^{-1} - \mathbf{R}^{-1} \mathbf{W} \mathbf{C}^{-1} \mathbf{W}' \mathbf{R}^{-1}) \mathbf{Z}_i)
\]
Define $S_i$ to be a $(t + b) \times b_i$ matrix so that $WS_i = Z_i$. This means that $S_i$ contains zeros everywhere except for an identity matrix of order $b_i$ in the partition corresponding to $Z_i$ in $W$. This allows us to write the above trace term as

$$
\text{tr}(PZ_iG_{ij}Z'_i) = \text{tr}(G_{ij}Z'_iPZ_i)
= \text{tr}(G_{ij}S'_iW'(R^{-1} - R^{-1}WC^{-1}W'R^{-1})WS_i)
= \text{tr}(G_{ij}S'_i(W'R^{-1}W - W'R^{-1}WC^{-1}W'R^{-1}W)S_i)
= \text{tr}(G_{ij}S'_iW'R^{-1}WC^{-1}(C - W'R^{-1}W)S_i)
$$

(2.37)

We can write the coefficient matrix $C$ in (2.12) as

$$
C = W'R^{-1}W + G^*
$$

where

$$
G^* = \begin{bmatrix}
0 & 0 \\
0 & G^{-1}
\end{bmatrix}
$$

and therefore

$$
W'R^{-1}W = C - G^*
$$

(2.38)

and rearranging gives

$$
C - W'R^{-1}W = G^*
$$

This allows the trace term in (2.37) to be written as

$$
\text{tr}(PZ_iG_{ij}Z'_i) = \text{tr}(G_{ij}S'_i(C - G^*)C^{-1}G^*S_i)
= \text{tr}(G_{ij}S'_iG^*S_i) - \text{tr}(G_{ij}S'_iG^*C^{-1}G^*S_i)
$$

(2.39)

We can write $G^*C^{-1}G^*$ as

$$
G^*C^{-1}G^* = \begin{bmatrix}
0 & 0 \\
0 & G^{-1}
\end{bmatrix}
\begin{bmatrix}
C^{XX} & C^{XZ} \\
C^{ZX} & C^{ZZ}
\end{bmatrix}
\begin{bmatrix}
0 & 0 \\
0 & G^{-1}
\end{bmatrix}
$$

$$
= \begin{bmatrix}
0 & 0 \\
0 & G^{-1}C^{ZZ}G^{-1}
\end{bmatrix}
$$

where $C^{ZZ}$ is the partition of $C^{-1}$ that corresponds to $u$ as defined in (2.20). Since $u = (u'_1 u'_2 \ldots u'_q)'$, $C^{ZZ}$ can also be partitioned as follows

$$
C^{ZZ} = \begin{bmatrix}
C^{Z_11} \quad C^{Z_1Z_2} \quad \cdots \quad C^{Z_1Z_q} \\
C^{Z_21} \quad C^{Z_2Z_2} \quad \cdots \quad C^{Z_2Z_q} \\
\vdots \quad \vdots \quad \ddots \quad \vdots \\
C^{Z_q1} \quad C^{Z_qZ_2} \quad \cdots \quad C^{Z_qZ_q}
\end{bmatrix}
$$

where $C^{Z_i1}$ is the partition of the inverse of $C$ that corresponds to the $i$th random effect, $u_i$. Noting that $G$ can also be partitioned as shown in (2.2), it follows that

$$
G^{-1}C^{ZZ}G^{-1} =
\begin{bmatrix}
G^{-1}_1 \quad G^{-1}_1C^{Z_1Z_1}G^{-1}_1 \quad \cdots \quad G^{-1}_1C^{Z_1Z_q}G^{-1}_q \\
G^{-1}_2 \quad G^{-1}_2C^{Z_2Z_1}G^{-1}_1 \quad \cdots \quad G^{-1}_2C^{Z_2Z_q}G^{-1}_q \\
\vdots \quad \vdots \quad \ddots \quad \vdots \\
G^{-1}_q \quad G^{-1}_qC^{Z_qZ_1}G^{-1}_1 \quad \cdots \quad G^{-1}_qC^{Z_qZ_q}G^{-1}_q
\end{bmatrix}
$$

2.2. ESTIMATION

23
and since $S_i$ contains zeros everywhere except for an identity matrix of order $b_i$ in the partition corresponding to $Z_i$ in $W$, it then follows that

$$S_i'G^1C^{-1}G^*S_i = G_i^{-1}C^{Z_iZ_i}G_i^{-1}$$  \hspace{1cm} (2.40)$$

Also note that

$$S_i'G^*S_i = G_i^{-1} \hspace{1cm} (2.41)$$

Equations (2.40) and (2.41) allow the trace term in (2.39) to be written as

$$\operatorname{tr}(PZ_i\dot{G}_{ij}Z_i') = \operatorname{tr}(\dot{G}_{ij}G_i^{-1}) - \operatorname{tr}(G_{ij}G_i^{-1}C^{Z_iZ_i}G_i^{-1}) \hspace{1cm} (2.42)$$

Now consider the second term in the REML score for $\gamma_{ij}$ in (2.36)

$$y'PZ_i\dot{G}_{ij}Z_i'Py = \dot{u}_i'G_i^{-1}\dot{G}_{ij}G_i^{-1}\dot{u}_i \hspace{1cm} (2.43)$$

since $\dot{u} = GZ'Py$. Combining (2.42) and (2.43) allows us to write the REML score for $\gamma_{ij}$ in (2.36) as

$$U(\gamma_{ij}) = -\frac{1}{2} \left( \operatorname{tr}(G_i^{-1}\dot{G}_{ij}) - \operatorname{tr}(C^{Z_iZ_i}\dot{G}_{ij}G_i^{-1}) - \frac{\dot{u}_i'G_i^{-1}\dot{G}_{ij}G_i^{-1}\dot{u}_i}{\theta} \right) \hspace{1cm} (2.44)$$

To estimate $\gamma_{ij}$ we need to equate the above equation to zero and solve for $\gamma_{ij}$. In general this cannot be done directly and an iterative scheme is required.

Note if there is only one random term then the REML score for $\gamma_i$ is given by

$$U(\gamma_i) = -\frac{1}{2} \left( \operatorname{tr}(G^{-1}\dot{G}_i) - \operatorname{tr}(C^{ZZg^{-1}\dot{G}_iG_i^{-1}) - \frac{\dot{u}_i'G_i^{-1}\dot{G}_iG_i^{-1}\dot{u}_i}{\theta} \right) \hspace{1cm} (2.45)$$

We can obtain the REML score for $\sigma^2$ by substituting $\dot{H}_i = \Sigma$ into (2.31).

$$U(\sigma^2) = -\frac{1}{2} \left( \operatorname{tr}(P\Sigma) - \frac{y'P\Sigma Py}{\theta} \right) \hspace{1cm} (2.46)$$

From Result A.5 we have

$$P = R^{-1} - R^{-1}W\Sigma^{-1}W'R^{-1} \quad = \frac{\Sigma^{-1}}{\sigma^2} \left( R^{-1}W\Sigma^{-1}W'R^{-1} \right)$$

and hence

$$PS = \frac{I_n - R^{-1}W\Sigma^{-1}W'W^1}{\sigma^2}$$
This allows us to write the first term in (2.51) as

\[
\text{tr}(P \Sigma) = \frac{\text{tr}(I_n - R^{-1}WC^{-1}W')}{\sigma^2} = \frac{n - \text{tr}(R^{-1}WC^{-1}W')}{\sigma^2} = \frac{n - \text{tr}(W'R^{-1}WC^{-1})}{\sigma^2} \tag{2.47}
\]

Recall from (2.16) that \( \hat{e} = RPy = \sigma^2 \Sigma Py \) and hence

\[
y'PSy = y'P \Sigma \Sigma^{-1} \Sigma Py = \frac{\hat{e}' \Sigma^{-1} \hat{e}}{\sigma^4}
\]

This allows the REML score for \( \sigma^2 \) in (2.46) to be written as

\[
U(\sigma^2) = -\frac{1}{2} \left( \frac{n - \text{tr}(W'R^{-1}WC^{-1})}{\sigma^2} - \frac{\hat{e}' \Sigma^{-1} \hat{e}}{\sigma^4} \right) \tag{2.48}
\]

Recall from (2.38) that \( W'R^{-1}W = C - G^* \) and hence

\[
\text{tr}(W'R^{-1}WC^{-1}) = \text{tr}((C - G^*)C^{-1}) = \text{tr}(I_{t+b} - G^*C^{-1}) = t + b - \text{tr}(G^*C^{-1}) \tag{2.49}
\]

Note that

\[
G^*C^{-1} = \begin{bmatrix}
0 & 0 \\
0 & G^*
\end{bmatrix} \begin{bmatrix}
C^{XX} & C^{XZ} \\
C^{ZX} & C^{ZZ}
\end{bmatrix} = \begin{bmatrix}
0 & G^{-1}C^{ZX} \\
G^{-1}C^{ZX} & C^{-1}C^{ZZ}
\end{bmatrix}
\]

therefore

\[
\text{tr}(G^*C^{-1}) = \text{tr}(G^{-1}C^{ZZ})
\]

We can now write the trace term in (2.49) as

\[
\text{tr}(W'R^{-1}WC^{-1}) = t + b - \text{tr}(G^{-1}C^{ZZ}) \tag{2.50}
\]

and hence the trace term in (2.47) is given by

\[
\text{tr}(P \Sigma) = \frac{n - t - b + \text{tr}(G^{-1}C^{ZZ})}{\sigma^2}
\]

We can now write the REML score for \( \sigma^2 \) in (2.48) as

\[
U(\sigma^2) = -\frac{1}{2} \left( \frac{n - t - b + \text{tr}(G^{-1}C^{ZZ})}{\sigma^2} - \frac{\hat{e}' \Sigma^{-1} \hat{e}}{\sigma^4} \right) \tag{2.51}
\]
CHAPTER 2. MIXED MODELS

Note that $C^Z$ and $\tilde{e}$ are functions of $\sigma^2$ and therefore we cannot find an algebraic solution for $\sigma^2$ and an iterative scheme is required.

We can obtain the REML score for $\phi_i$ by substituting $\dot{H}_i = \sigma^2 \dot{\Sigma}_i$ into (2.31).

$$U(\phi_i) = -\frac{1}{2} \left( \sigma^2 \text{tr}(P\Sigma) - \frac{\sigma^2 y' P \dot{\Sigma}_i P y}{\theta} \right)$$ (2.52)

From Result A.5 we know that

$$P = R^{-1} - R^{-1} WC^{-1} W'R^{-1}$$

and hence

$$\text{tr}(P \dot{\Sigma}_i) = \text{tr} \left( \Sigma^{-1} \dot{\Sigma}_i \right) = \frac{\text{tr}(\Sigma^{-1} \dot{\Sigma}_i) - \text{tr}(C^{-1} W' \Sigma^{-1} \dot{\Sigma}_i \Sigma^{-1} W)}{\sigma^4}$$

We can now write the first term in (2.52) as

$$\sigma^2 \text{tr}(P \dot{\Sigma}_i) = \text{tr}(\Sigma^{-1} \dot{\Sigma}_i) - \frac{\text{tr}(C^{-1} W' \Sigma^{-1} \dot{\Sigma}_i \Sigma^{-1} W)}{\sigma^2}$$

Recall from (2.16) that $\tilde{e} = RPy = \sigma^2 \Sigma Py$. Therefore

$$y' P \dot{\Sigma}_i P y = y' P \Sigma \Sigma^{-1} \dot{\Sigma}_i \Sigma^{-1} \Sigma P y = \frac{\tilde{e}' \Sigma^{-1} \dot{\Sigma}_i \Sigma^{-1} \tilde{e}}{\sigma^4}$$

and hence the second term in (2.52) can be written as

$$\frac{\sigma^2 y' P \dot{\Sigma}_i P y}{\theta} = \frac{\tilde{e}' \Sigma^{-1} \dot{\Sigma}_i \Sigma^{-1} \tilde{e}}{\sigma^2 \theta}$$

We can now write the REML score for $\phi_i$ in (2.52) as

$$U(\phi_i) = -\frac{1}{2} \left( \text{tr}(\Sigma^{-1} \dot{\Sigma}_i) - \frac{\text{tr}(C^{-1} W' \Sigma^{-1} \dot{\Sigma}_i \Sigma^{-1} W)}{\sigma^2} - \frac{\tilde{e}' \Sigma^{-1} \dot{\Sigma}_i \Sigma^{-1} \tilde{e}}{\sigma^2 \theta} \right)$$ (2.53)

To estimate $\phi_i$ we equate to zero and solve for $\phi_i$. In general this cannot be solved directly and an iterative scheme is required.

In summary the variance parameters to be estimated are $\theta$ and $\kappa$. The REML estimate for $\theta$ is given in (2.29). Given an estimate of $\kappa = \kappa^{(m)}$ an update of $\theta$ is given by

$$\theta^{(m)} = \frac{y' P^{(m)} y}{n - t}$$ (2.54)

where $P^{(m)}$ is $P$ evaluated at $\kappa = \kappa^{(m)}$. In general an iterative scheme is required to estimate $\kappa = (\gamma', \sigma^2, \phi')'$. 
2.3 Iterative Schemes

In general the REML log-likelihood function cannot be maximised analytically with respect to the variance parameters, and numerical approaches must be used. In principle, the solutions can be obtained by performing an exhaustive grid search, numerically evaluating the REML log-likelihood at each point on a grid covering the entire range of the parameter space. The solution is then defined by the point on the grid that gives the largest REML log-likelihood. However this procedure is impractical if there are more than a few variance parameters, since each variance parameter adds to the dimensionality of the search.

Various iterative techniques for solving ML/REML equations have been proposed. Iterative methods all have a common structure. All iterative methods require initial starting values for the variance parameters. Beginning with starting values, there is a rule for calculating the value of the parameter for the next iteration, and a rule for deciding when to stop iterating and declare the current value to be the REML estimate (Thisted (1988)). A good iterative method should converge to a global maximum for a wide range of starting values, at each iteration it should be relatively quick to calculate the update, and it should converge in relatively few iterations (Searle et al., 1992). A potential problem with all iterative methods is that they may not converge to the true REML estimates if the log-likelihood surface contains several local maxima. Hence when applying iterative methods, several sets of starting values should be used.

A wide variety of iterative techniques have been proposed based on various modifications of two basic approaches: the EM algorithm and the Newton-Raphson algorithm. In the following chapters (3, 4 and 5) we develop three algorithms for obtaining REML estimates of the variance parameters; the AI algorithm (a variant of the Newton-Raphson algorithm), the EM algorithm and the PXEM algorithm (a variant of the EM algorithm). The choice of stopping criteria will be discussed in Chapter 6 and the selection of starting values will be discussed in Chapter 7.
Chapter 3

AI Algorithm

3.1 Introduction

In their original description of REML estimation, Patterson and Thompson (1971) used the Fisher Scoring (FS) algorithm to solve the REML score equations. Given estimates of $\theta = \theta^{(m)}$ and $\kappa = \kappa^{(m)}$, updates of $\theta = \theta^{(m+1)}$ and $\kappa = \kappa^{(m+1)}$ using the FS algorithm are given by

$$
\begin{bmatrix}
\theta^{(m+1)} \\
\kappa^{(m+1)}
\end{bmatrix}
= \begin{bmatrix}
\theta^{(m)} \\
\kappa^{(m)}
\end{bmatrix}
+ \mathcal{I}_E^{(m)-1} \begin{bmatrix}
U(\theta^{(m)}) \\
U(\kappa^{(m)})
\end{bmatrix}
$$

(3.1)

where $U(\theta)$ is the REML score for $\theta$ in (2.28) and $U(\kappa^{(m)}) = (U(\kappa_1), U(\kappa_2), \ldots, U(\kappa_i))'$ is the score vector for $\kappa$ where $U(\kappa_i)$ is the REML score for $\kappa_i$ given in (2.31). $\mathcal{I}_E$ is the expected information matrix

$$
\mathcal{I}_E = \begin{bmatrix}
\mathcal{I}_E(\theta, \theta) & \mathcal{I}_E(\theta, \kappa) \\
\mathcal{I}_E(\kappa, \theta) & \mathcal{I}_E(\kappa, \kappa)
\end{bmatrix}
$$

Define the elements of the inverse of the expected information matrix, $\mathcal{I}_E^{-1}$, to be

$$
\mathcal{I}_E^{-1} = \begin{bmatrix}
\mathcal{I}_E^{(\theta, \theta)} & \mathcal{I}_E^{(\theta, \kappa)} \\
\mathcal{I}_E^{(\kappa, \theta)} & \mathcal{I}_E^{(\kappa, \kappa)}
\end{bmatrix}
$$

$\mathcal{I}_E^{(m)-1}$ is the inverse of the expected information matrix evaluated at $\theta^{(m)}$ and $\kappa^{(m)}$.

Recall from Chapter 2 that we have an algebraic solution for $\theta$, as given in (2.54), and hence we only need an update for $\kappa$. From (3.1) the FS update for $\kappa$ is given by

$$
\kappa^{(m+1)} = \kappa^{(m)} + \mathcal{I}_E^{(\kappa^{(m)}, \theta^{(m)})} U(\theta^{(m)}) + \mathcal{I}_E^{(\kappa^{(m)}, \kappa^{(m)})} U(\kappa^{(m)})
$$

Evaluating the REML score for $\theta$ given in (2.28) at the REML estimate for $\theta^{(m)} = y' P^{(m)} y / (n - t)$ as given in (2.54), we get

$$
U(\theta^{(m)}) = -\frac{1}{2} \left( \frac{1}{y' P^{(m)} y} \frac{(n - t)}{y' P^{(m)} y} - y' P^{(m)} y \frac{(n - t)^2}{(y' P^{(m)} y)^2} \right) = 0
$$
and hence the FS update for $\kappa$ is given by

$$
\kappa^{(m+1)} = \kappa^{(m)} + I_E^{(\kappa^{(m)}, \kappa^{(m)})} U(\kappa^{(m)})
$$

(3.2)

where $I_E^{(\kappa^{(m)}, \kappa^{(m)})}$ is the partition of $I_E^{(m)-1}$ relevant to $\kappa$.

In the case when $\theta = 1$ the expected information matrix is given by $I_E = I_E(\kappa, \kappa)$ and the update is given by

$$
\kappa^{(m+1)} = \kappa^{(m)} + I_E(\kappa^{(m)}, \kappa^{(m)})^{-1} U(\kappa^{(m)})
$$

where $I_E(\kappa^{(m)}, \kappa^{(m)})^{-1}$ is the inverse of the expected information matrix evaluated at $\kappa^{(m)}$.

In some cases as discussed by Thompson (1977), the observed information matrix $I_O$ is easier to calculate than the expected information matrix $I_E$ and so it is used in place of $I_E$ in (3.2). This algorithm is known as the Newton-Raphson (NR) algorithm.

To use the FS and NR algorithms we need the observed and expected information matrices.

The elements of the observed information matrix, $I_O$, are given by

$$
I_O(\theta, \theta) = -\frac{\partial^2 \ell_R(\theta, \kappa; y_2)}{\partial^2 \theta^2}
= -\frac{\partial U(\theta)}{\partial \theta}
= \frac{y' P y}{(\theta)^3} - \frac{(n-1)}{2(\theta)^2}
$$

$$
I_O(\theta, \kappa_i) = -\frac{\partial^2 \ell_R(\theta, \kappa; y_2)}{\partial \theta \partial \kappa_i}
= -\frac{\partial U(\theta)}{\partial \kappa_i}
= \frac{y' PH_i P y}{2(\theta)^2}
$$

$$
I_O(\kappa_i, \kappa_j) = -\frac{\partial^2 \ell_R(\theta, \kappa; y_2)}{\partial \kappa_i \partial \kappa_j}
= -\frac{\partial U(\kappa_i)}{\partial \kappa_j}
= \frac{1}{2} \left( \frac{\partial \text{tr}(PH_i)}{\partial \kappa_j} - \frac{1}{\theta} \frac{\partial y' P H_i P y}{\partial \kappa_j} \right)
$$

(3.3)
Define $\ddot{H}_{ij} = \frac{\partial H_i}{\partial \kappa_j} = \frac{\partial^2 H}{\partial \kappa_i \partial \kappa_j}$ and consider

$$
\frac{\partial \text{tr}(P \dot{H}_i)}{\partial \kappa_j} = \text{tr} \left( \frac{\partial P \dot{H}_i}{\partial \kappa_j} \right)
= \text{tr} \left( \frac{\partial P H_i}{\partial \kappa_j} + \frac{\partial \dot{H}_i}{\partial \kappa_j} P \right)
= \text{tr}(-P \dot{H}_j P \dot{H}_i + P \ddot{H}_i)
= \text{tr}(P \ddot{H}_i) - \text{tr}(P \dot{H}_j P \dot{H}_j)
$$

and

$$
\frac{\partial y' P \dot{H}_i P y}{\partial \kappa_j} = y' \frac{\partial P \dot{H}_i P}{\partial \kappa_j} y
= y' \left( \frac{\partial P \dot{H}_i P}{\partial \kappa_j} + P \frac{\partial \dot{H}_i P}{\partial \kappa_j} + P \frac{\partial P}{\partial \kappa_j} \right) y
= y'(-P \dot{H}_j P \dot{H}_i P + P \ddot{H}_i P - P \ddot{H}_i P \dot{H}_j) y
= y'P \ddot{H}_i P y - 2 y'P \dot{H}_i P \dot{H}_j P y
$$

Hence $I_O(\kappa_i, \kappa_j)$ in (3.3) can be written as

$$
I_O(\kappa_i, \kappa_j) = \frac{1}{2} \left( \text{tr}(P \ddot{H}_i) - \text{tr}(P \dot{H}_i P \dot{H}_j) - \frac{y' P \ddot{H}_i P y}{\theta} + \frac{2 y' P \ddot{H}_i P \dot{H}_j P y}{\theta} \right)
$$

The elements of the expected information matrix, $I_E$ are given by

$$
I_E(\theta, \theta) = E \left[ -\frac{\partial^2 \ell_R(\theta, \kappa; y_2)}{\theta^2} \right]
= E[I_o(\theta, \theta)]
= E[y' P y] - \frac{(n - t)}{2 \theta^2} \tag{3.4}
$$

$$
I_E(\theta, \kappa_i) = E \left[ -\frac{\partial^2 \ell_R(\theta, \kappa; y_2)}{\theta \partial \kappa_i} \right]
= E[I_o(\theta, \kappa_i)]
= E[y' P \dot{H}_i P y] \frac{1}{2 \theta^2} \tag{3.5}
$$

$$
I_E(\kappa_i, \kappa_j) = E \left[ -\frac{\partial^2 \ell_R(\theta, \kappa; y_2)}{\partial \kappa_i \partial \kappa_j} \right]
= E[I_o(\kappa_i, \kappa_j)]
= \frac{1}{2} \left( \text{tr}(P \ddot{H}_i) - \text{tr}(P \dot{H}_i P \dot{H}_j) - \frac{E[y' P \ddot{H}_i P]}{\theta} \right)
+ \frac{2 E[y' P \dot{H}_i P \dot{H}_j P]}{\theta} \tag{3.6}
$$
Using Result A.10 we have

\[ E[yPy] = \theta \text{tr}(HP) + \tau'X'PX\tau \]  \hspace{1cm} (3.7)

\[ E[y'PH_iPy] = \theta \text{tr}(HP\dot{H}_iP) + \tau'X'PH_iPX\tau \]  \hspace{1cm} (3.8)

\[ E[y'PH_{ij}Py] = \theta \text{tr}(HP\ddot{H}_{ij}P) + \tau'X'PH_{ij}PX\tau \]  \hspace{1cm} (3.9)

\[ E[y'PH_iPH_jPy] = \theta \text{tr}(HP\dot{H}_iPH_jP) + \tau'X'PH_iPH_jPX\tau \]  \hspace{1cm} (3.10)

The trace term in (3.7) can be written as

\[ \text{tr}(HP) = \text{tr}(H(H^{-1} - H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1})) \]
\[ = \text{tr}(I_n - X(X'H^{-1}X)^{-1}X'H^{-1}) \]
\[ = \text{tr}(I_n) - \text{tr}(X(X'H^{-1}X)^{-1}X'H^{-1}) \]
\[ = n - \text{tr}((X'H^{-1}X)^{-1}X'H^{-1}X) \]
\[ = n - \text{tr}(I_t) \]
\[ = n - t \]

The trace term in (3.8) can be written as

\[ \text{tr}(HP\dot{H}_iP) = \text{tr}(PHP\dot{H}_i) \]
\[ = \text{tr}(PH_i) \]

The trace term in (3.9) can be written as

\[ \text{tr}(HP\ddot{H}_{ij}P) = \text{tr}(PHP\ddot{H}_{ij}) \]
\[ = \text{tr}(PH_{ij}) \]

And the trace term in (3.10) can be written as

\[ \text{tr}(HP\dot{H}_iPH_jP) = \text{tr}(PHP\dot{H}_iPH_j) \]
\[ = \text{tr}(PH_iPH_j) \]

Noting that

\[ PX\tau = H^{-1}X\tau - H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1}X\tau \]
\[ = H^{-1}X\tau - H^{-1}X\tau \]
\[ = 0 \]

allows us to write the expectations in (3.7), (3.8), (3.9) and (3.10) as

\[ E[yPy] = \theta(n - t) \]
\[ E[y'PH_iPy] = \theta\text{tr}(PH_i) \]
\[ E[y'PH_{ij}Py] = \theta\text{tr}(PH_{ij}) \]
\[ E[y'PH_iPH_jPy] = \theta\text{tr}(PH_iPH_j) \]
We can now write the elements of the expected information matrix in (3.4), (3.5) and (3.6) as
\[ I_E(\theta, \theta) = \frac{(n-t)}{2\theta^2} \]
\[ I_E(\theta, \kappa_i) = \frac{\text{tr}(P\dot{H}_i)}{2\theta} \]
\[ I_E(\kappa_i, \kappa_j) = \frac{\text{tr}(P\dot{H}_iP\dot{H}_j)}{2} \]

### 3.2 AI Algorithm

For large data sets, the evaluation of some of the above traces involving matrices of order \( n \times n \) is either not feasible or very computer intensive. Various computational methods have been presented to overcome this problem. Graser et al. (1987) suggest a derivative free (DF) algorithm that makes use of the Nelder and Mead (1965) simplex algorithm. This procedure is useful when there are only a few variance parameters but the rate of convergence decreases as the number of variance parameters increases. Dempster et al. (1977) propose the EM algorithm as a general method of estimation that can be used to estimate variance parameters in linear mixed models. The EM algorithm will be discussed later in Chapter 4. Gilmour et al. (1995) overcome the problem of evaluating traces of large matrices by considering the average information matrix which is a simplified average of the terms in the observed and expected information matrices.

The elements of the average information matrix \( I_A \) are given by
\[ I_A(\theta, \theta) = \frac{y'Py}{2\theta^3} \]
\[ I_A(\theta, \kappa_i) = \frac{y'P\dot{H}_iPy}{2\theta^2} \]
\[ I_A(\kappa_i, \kappa_j) = \frac{y'P\dot{H}_iP\dot{H}_jPy}{2\theta} \]

The first element is the average of \( I_O(\theta, \theta) \) and \( I_E(\theta, \theta) \). The second element is the average of \( I_O(\theta, \kappa_i) \) and \( I_E(\theta, \kappa_i) \) where \( \text{tr}(P\dot{H}_i) \) is approximated by \( \frac{y'P\dot{H}_iPy}{\theta} \), since \( E[y'P\dot{H}_iPy] = \theta \text{tr}(P\dot{H}_i) \). The third element is the average of \( I_O(\kappa_i, \kappa_j) \) and \( I_E(\kappa_i, \kappa_j) \) where \( \text{tr}(P\ddot{H}_{ij}) \) is approximated by \( \frac{y'P\ddot{H}_{ij}Py}{\theta} \), since \( E[y'P\ddot{H}_{ij}Py] = \theta \text{tr}(P\ddot{H}_{ij}) \).

For variance components models, that is models in which the variance structure is linear in the parameters, \( \ddot{H}_{ij} = 0 \). And therefore \( I_A(\kappa_i, \kappa_j) \) is an exact average of \( I_O(\kappa_i, \kappa_j) \) and \( I_E(\kappa_i, \kappa_j) \).

Note that the REML score for \( \kappa_i \) in (2.31) is composed of 2 parts; a trace term \( -\frac{1}{2}\text{tr}(P\dot{H}_i) \) and a data part \( \frac{y'P\dot{H}_iPy}{2\theta} \). When we have a linear variance model, \( I_A(\kappa_i, \kappa_j) \) can be found...
by taking the derivative of the data part of the REML score for $\kappa_i$ and dividing it by $-2$.

\[
- \frac{1}{2} \frac{\partial}{\partial \kappa_j} \left( \frac{y' P \dot{H}_j P y}{2\theta} \right) = - \frac{1}{4\theta} \left( 2y' P \dot{H}_i \frac{\partial P}{\partial \kappa_j} y + y' P \frac{\partial \dot{H}_i}{\partial \kappa_j} P y \right)
\]

\[
= - \frac{1}{4\theta} \left( -2y' P \dot{H}_i P \dot{H}_j P y + y' P \ddot{H}_{ij} P y \right)
\]

\[
= \frac{y' P \dot{H}_i P \dot{H}_j}{2\theta} (\text{since } \ddot{H}_{ij} = 0)
\]

\[
= I_A(\kappa_i, \kappa_j)
\]

The elements of $I_A$ are much easier to calculate than the elements of $I_O$ and $I_E$ as they do not involve traces. The average information matrix is used in place of the expected information matrix in the FS algorithm. This algorithm is called the average information (AI) algorithm and is implemented in the ASReml software (Gilmour et al., 2006). An outline of the computing strategy is given in the next section.

Noting that $PHP = P$ as proven in Result A.6, the elements of $I_A$ can be written as

\[
I_A(\theta, \theta) = \frac{1}{2\theta} \left( \frac{HPy}{\theta} \right)' P \left( \frac{HPy}{\theta} \right) = \frac{1}{2\theta} q_\theta^T P q_\theta
\]

\[
I_A(\theta, \kappa_i) = \frac{1}{2\theta} \left( \frac{HPy}{\theta} \right)' P \left( \dot{H}_i P y \right) = \frac{1}{2\theta} q_\theta^T P q_{\kappa_i}
\]

\[
I_A(\kappa_i, \kappa_j) = \frac{1}{2\theta} \left( \dot{H}_i P y \right)' P \left( \dot{H}_j P y \right) = \frac{1}{2\theta} q_{\kappa_i}^T P q_{\kappa_j}
\]

where $q_\theta$ and $q_{\kappa_i}$ are the “working variables” for $\theta$ and $\kappa_i$, $i = 1, \ldots, k$, given by

\[
q_\theta = \frac{HPy}{\theta}
\]

\[
q_{\kappa_i} = \frac{\dot{H}_i P y}{\theta}
\]

If we define the matrix $Q = [ q_\theta \quad q_{\kappa_1} \quad \ldots \quad q_{\kappa_k} ]$ then the matrix $I_A$ is a scaled residual sums of squares matrix given by

\[
I_A = \frac{1}{2\theta} Q' PQ
\]

as discussed in Gilmour et al. (1995).

For the scale parameter $\theta$, the working variable can be written as

\[
q_\theta = \frac{HPy}{\theta}
\]

\[
= \frac{(ZGZ' + R)y}{\theta}
\]

\[
= \frac{ZGZ' P y + RP y}{\theta}
\]

\[
= \frac{Z\hat{u} + \hat{e}}{\theta}
\]

\[ (3.13) \]
For a random effects variance parameter \( \gamma_{ij} \), \( \dot{H}_{ij} = Z_i \dot{G}_{ij} Z'_i \) as given in (2.32) and hence the working variable for \( \gamma_{ij} \) is given by

\[
q_{\gamma_{ij}} = Z_i \dot{G}_{ij} Z'_i \PY
\]

If there is only one random term then \( \dot{H}_i = Z \dot{G}_i Z \) as given in (2.33) and hence the working variable for \( \gamma_i \) is given by

\[
q_{\gamma_i} = Z \dot{G}_i G^{-1} \tilde{u}_i
\]

For the residual variance parameter \( \sigma^2 \), \( \dot{H}_i = \Sigma \) as given in (2.34) and hence the working variable for \( \sigma^2 \) is given by

\[
q_{\sigma^2} = \Sigma \PY = \frac{\dot{\epsilon}}{\sigma^2}
\]

For the error variance parameter \( \phi_i \), \( \dot{H}_i = \sigma^2 \tilde{\Sigma}_i \) as given in (2.35) and hence the working variable for \( \phi_i \) is given by

\[
q_{\phi_i} = \sigma^2 \tilde{\Sigma}_i \PY = \tilde{\Sigma}_i \Sigma^{-1} \dot{\epsilon}
\]

### 3.3 Absorption, Backsubstitution and Inversion

Recall from Chapter 2 that \( C^{-1} \) is required to obtain \( \hat{\beta} \) to solve the mixed model equations, the REML log-likelihood requires \( |C| \) and \( y'PY \), and the REML score equations require \( C^{-1} \). Gilmour et al. (1995) describe a recursive process of absorption, backsubstitution and inversion of the mixed model equations, that gives \( \hat{\beta}, C^{XX} \) and \( y'PY \), which then allow us to find \( C^{-1} \) and \( |C| \) easily. They show that this method is computationally convenient and efficient, particularly for large data sets. The absorption process is essentially Gaussian Elimination (Golub and van Loan (1983)) and the final step gives the residual sum of squares \( y'PY \). Backsubstitution determines \( \hat{\beta} \) using terms evaluated in the absorption process, and inversion produces \( C^{XX} \). The details are as follows.

The coefficient matrix of the mixed model equations, \( C \), is augmented to form

\[
M = \begin{bmatrix}
y' \dot{R}^{-1} y & y' \dot{R}^{-1} X & y' \dot{R}^{-1} Z \\
X' \dot{R}^{-1} y & X' \dot{R}^{-1} X & X' \dot{R}^{-1} Z \\
Z' \dot{R}^{-1} y & Z' \dot{R}^{-1} X & Z' \dot{R}^{-1} Z + G^{-1} \\
a & b'_x & b'_z \\
b_x & \tilde{C}_{XX} & \tilde{C}_{XZ} \\
b_z & \tilde{C}_{ZX} & \tilde{C}_{ZZ}
\end{bmatrix}
\]
where \( a = y'Py \) and \( b = \begin{bmatrix} b_X \\ b_z \end{bmatrix} = \begin{bmatrix} X'\mathbf{R}^{-1}y \\ Z'\mathbf{R}^{-1}y \end{bmatrix} = \mathbf{W}'\mathbf{R}^{-1}y \). Note that \( M \) is symmetric and hence can be stored as lower triangular.

At the start of the absorption process \( M \) is completely unabsorbed. In the absorption process each row of \( M \) is successively absorbed, from bottom to top row-wise, in correspondence with the diagonal elements of \( C \). We illustrate by absorbing partitions of \( \text{vec}C \) rather than single rows.

\( C_{ZZ} \) is absorbed by constructing

\[
\begin{align*}
M^* &= \begin{bmatrix} a & b'_X \\ b_X & C_{XX} \end{bmatrix} - \begin{bmatrix} b'_Z \\ C_{XZ} \end{bmatrix} C_{ZZ}^{-1} \begin{bmatrix} b_Z & C_{ZX} \end{bmatrix} \\
&= \begin{bmatrix} a - b'_Z C_{ZZ}^{-1} b_Z & b'_X - b'_Z C_{ZZ}^{-1} C_{ZX} \\ b_X - C_{XZ} C_{ZZ}^{-1} b_Z & C_{XX} - C_{XZ} C_{ZZ}^{-1} C_{ZX} \end{bmatrix} \\
&= \begin{bmatrix} M^*_{11} & M^*_{12} \\ M^*_{21} & M^*_{22} \end{bmatrix}
\end{align*}
\]

Considering the elements of \( M^* \) individually gives

\[
\begin{align*}
M^*_{11} &= a - b'_Z C_{ZZ}^{-1} b_Z \\
&= y'\mathbf{R}^{-1}y - y'\mathbf{R}^{-1}Z(Z'\mathbf{R}^{-1}Z + G^{-1})^{-1}Z'\mathbf{R}^{-1}y \\
&= y'(\mathbf{R}^{-1} - \mathbf{R}^{-1}Z(Z'\mathbf{R}^{-1}Z + G^{-1})^{-1}Z'\mathbf{R}^{-1})y \\
&= y'(\mathbf{R} + ZGZ)^{-1}y \quad \text{(using Result A.3)} \\
&= y'\mathbf{H}^{-1}y \\
M^*_{12} &= b'_X - b'_Z C_{ZZ}^{-1} C_{ZX} \\
&= y'\mathbf{R}^{-1}X - y'\mathbf{R}^{-1}Z(Z'\mathbf{R}^{-1}Z + G^{-1})^{-1}Z'\mathbf{R}^{-1}X \\
&= y'\mathbf{H}^{-1}X \\
M^*_{21} &= b'_X - C_{XZ} C_{ZZ}^{-1} b_Z \\
&= X'\mathbf{R}^{-1}y - X'\mathbf{R}^{-1}Z(Z'\mathbf{R}^{-1}Z + G^{-1})^{-1}Z'\mathbf{R}^{-1}y \\
&= X'\mathbf{H}^{-1}y \\
M^*_{22} &= C_{XX} - C_{XZ} C_{ZZ}^{-1} C_{ZZ} \\
&= X'\mathbf{H}^{-1}X \quad \text{(using (2.15))}
\end{align*}
\]

Therefore \( M^* \) is given by

\[
M^* = \begin{bmatrix} y'\mathbf{H}^{-1}y & y'\mathbf{H}^{-1}X \\ X'\mathbf{H}^{-1}y & X'\mathbf{H}^{-1}X \end{bmatrix}
\]

This completes the first stage of absorption.

Now let

\[
M^* = \begin{bmatrix} a^* & b'_X \\ b'_X & C_{XX}^* \end{bmatrix}
\]
where \( a^* = y'H^{-1}y \), \( b_X^* = X'H^{-1}y \) and \( C_{XX}^* = X'H^{-1}X \). Absorb \( C_{XX}^* \) by constructing

\[
M^{**} = a^* - b_X'(C_{XX}^*)^{-1}b_X^*
\]
\[
= y'H^{-1}y - y'H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1}y
\]
\[
= y'(H^{-1} - H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1})y
\]
\[
= y'Py
\]

This is the final step in the absorption process and it gives the residual sum of squares \( y'Py \).

We now need to carry out backsubstitution and inversion to obtain \( \hat{\beta} \) and \( C_{XX} \). \( M^* \) defines the set of equations

\[
X'H^{-1}X\hat{\tau} = X'H^{-1}y
\]
\[
(C_{XX} - C_{XZ}C_{ZZ}^{-1}C_{ZX})\hat{\tau} = X'H^{-1}y
\]

Solving for \( \hat{\tau} \) gives

\[
\hat{\tau} = (C_{XX} - C_{XZ}C_{ZZ}^{-1}C_{ZX})^{-1}X'H^{-1}y
\]

Using Result A.12 we know that \( C_{XX} = (C_{XX} - C_{XZ}C_{ZZ}^{-1}C_{ZX})^{-1} \). And from (2.15) we know that \( C_{XX} - C_{XZ}C_{ZZ}^{-1}C_{ZX} = X'H^{-1}X \). Thus we have

\[
C_{XX} = (X'H^{-1}X)^{-1}
\]

It follows that

\[
\hat{\tau} = (X'H^{-1}X)^{-1}X'H^{-1}y
\]

This is the BLUE of \( \tau \) as given in (2.6).

\( M \) defines the mixed model equations

\[
C_{XX}\hat{\tau} + C_{XZ}\hat{u} = b_X
\]
\[
C_{ZX}\hat{\tau} + C_{ZZ}\hat{u} = b_Z
\]

Substituting \( \hat{\tau} \) into the second set of equations gives

\[
\hat{u} = C_{ZZ}^{-1}(b_Z - C_{ZX}\hat{\tau})
\]
\[
= (Z'R^{-1}Z + G^{-1})^{-1}(Z'R^{-1}y - Z'R^{-1}X(X'H^{-1}X)^{-1}X'H^{-1}y)
\]
\[
= (Z'R^{-1}Z + G^{-1})^{-1}Z'R^{-1}(I - X(X'H^{-1}X)^{-1}X'H^{-1})y
\]
\[
= (Z'R^{-1}Z + G^{-1})^{-1}Z'R^{-1}HPy
\]
\[
= GZ'(R + ZGZ')^{-1}HPy \quad \text{(using Result A.3)}
\]
\[
= GZ'H^{-1}HPy
\]
\[
= GZ'Py
\]

This is the BLUP of \( u \) as given in (2.7).
Now that $\hat{\beta} = \begin{bmatrix} \hat{\tau} \\ \hat{u} \end{bmatrix}$, and $C^{XX}$ are available, the recursive process of absorption, back-substitution and inversion is complete.

The other elements of $C^{-1}$ can be calculated using Result A.12. The determinant of $C$ can be calculated using Result A.2

$$|C| = |C_{ZZ}| C_{XX} - C_{XZ} C_{ZZ}^{-1} C_{ZX}$$

where $C_{ZZ}$ and $M_{22}^*$ are the pivots in the absorption process.

Absorption is clearly a computationally convenient and efficient way of determining $y'yPy$ as it also gives results for $\hat{\beta}$ and $C^{XX}$ which can be used to find $C^{-1}$ and $|C|$, all of which are needed to solve the mixed model equations, the residual log-likelihood and the REML score equations.

We know from (3.12) that the average information matrix can be written as

$$\mathcal{I}_A = \frac{1}{2\theta} Q'PQ$$

where $Q = [ q_q q_{s_1} \ldots q_{s_k} ]$, $q_q = \frac{HP_q}{\sigma}$ and $q_{s_i} = \hat{H}_iPy$ for $i = 1, \ldots, k$. $Q'PQ$ is of the same form as $y'yPy$ which was obtained via absorption of the matrix $M$. This suggests that $Q'PQ$ could be obtained by performing absorption of $M$ but with the data vector $y$ replaced with the matrix $Q$.

Augment $C$ to form

$$M = \begin{bmatrix} Q'R^{-1}Q & Q'R^{-1}X & Q'R^{-1}Z \\ X'R^{-1}Q & X'R^{-1}X & X'R^{-1}Z \\ Z'R^{-1}Q & Z'R^{-1}X & Z'R^{-1}Z + G^{-1} \end{bmatrix} = \begin{bmatrix} A & b'_X & b'_Z \\ b_X & C_{XX} & C_{XZ} \\ b_Z & C_{ZX} & C_{ZZ} \end{bmatrix}$$

where $A = Q'R^{-1}Q$, $b_X = X'R^{-1}Q$ and $b_Z = Z'R^{-1}Q$. Absorb $C_{ZZ}$ by constructing

$$M^* = \begin{bmatrix} A & b'_X & b'_Z \\ b_X & C_{XX} & C_{XZ} \end{bmatrix} - \begin{bmatrix} b'_Z \\ C_{XZ} \end{bmatrix} C_{ZZ}^{-1} \begin{bmatrix} b_Z & C_{ZZ} \end{bmatrix} = \begin{bmatrix} A - b'_Z C_{ZZ}^{-1} b_Z & b'_X - b'_Z C_{ZZ}^{-1} C_{XX} \\ b_X - C_{XX} C_{ZZ}^{-1} b_Z & C_{XX} - C_{XX} C_{ZZ}^{-1} C_{XX} \end{bmatrix}$$
Considering the elements of $M^*$ separately gives

$$M^*_{11} = A - b'_Z C_{zz}^{-1} b_Z$$
$$= Q'R^{-1}Q - Q'R^{-1}Z(Z'R^{-1}Z + G^{-1})^{-1}Z'R^{-1}Q$$
$$= Q'(R^{-1} - R^{-1}Z(Z'R^{-1}Z + G^{-1})^{-1}Z'R^{-1})Q$$
$$= Q'(R + ZGZ')^{-1}Q \quad \text{(using Result A.3)}$$
$$= Q'H^{-1}Q$$

$$M^*_{12} = b'_X - b'_Z C_{zz}^{-1} C_{xz}$$
$$= Q'R^{-1}X - Q'R^{-1}Z(Z'R^{-1}Z + G^{-1})^{-1}Z'R^{-1}X$$
$$= Q'H^{-1}X$$

$$M^*_{21} = b_X - C_{xz} C_{zz}^{-1} b_Z$$
$$= X'R^{-1}Q - X'R^{-1}Z(Z'R^{-1}Z + G^{-1})^{-1}Z'R^{-1}Q$$
$$= X'H^{-1}Q$$

$$M^*_{22} = C_{xx} - C_{xz} C_{zz}^{-1} C_{zx}$$
$$= X'H^{-1}X \quad \text{(using (2.15))}$$

Therefore $M^*$ is given by

$$M^* = \begin{bmatrix}
Q'H^{-1}Q & Q'H^{-1}X \\
X'H^{-1}Q & X'H^{-1}X
\end{bmatrix}$$

This completes the first step of the absorption process.

Now let

$$M^* = \begin{bmatrix}
A^* & b'_X \\
b'_X & C_{xx}^*
\end{bmatrix}$$

where $A^* = Q'H^{-1}Q$, $b^*_X = X'H^{-1}Q$ and $C_{xx}^* = X'H^{-1}X$. Absorb $C_{xx}^*$ by constructing

$$M^{**} = A^* - b'_X (C_{xx}^*)^{-1} b'_X$$
$$= Q'H^{-1}Q - Q'H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1}Q$$
$$= Q'(H^{-1} - H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1})Q$$
$$= Q'PQ$$

This is the final step in the absorption process and it gives $Q'PQ$, thus demonstrating that absorption can be used to determine the average information matrix in (3.12).
Chapter 4

EM Algorithm

4.1 Introduction

An alternative method to the AI algorithm for finding estimates of the variance parameters is the EM algorithm. Dempster et al. (1977) proposed the EM (Expectation - Maximisation) algorithm as a general approach to iterative computation of maximum likelihood estimates for incomplete data problems. There may be missing, truncated or censored data. In other situations the data may not be evidently incomplete, but the statistician may be able to formulate the incompleteness in such a way that facilitates the use of the EM algorithm.

Dempster et al. (1977) view the observed data $y_o$ as incomplete. The complete data $y_c$ is made up of the observed data $y_o$ and the unobserved or missing data denoted by $y_m$, that is $y_c = (y_o', y_m')'$. Let the density function of the observed data be $g(y_o; \kappa)$, where $\kappa$ is a vector of unknown parameters. Denote the density function of the complete data by $g_c(y_c; \kappa) = g_c(y_o, y_m; \kappa)$. The density function of the incomplete data can be written in terms of the density function of the complete data as

$$g(y_o; \kappa) = \int g_c(y_o, y_m; \kappa) \, dy_m$$  \hspace{1cm} (4.1)

Denote the log-likelihood function for the observed data by

$$\ell(\kappa; y_o) = \log g(y_o; \kappa)$$

and the log-likelihood function for the complete data by

$$\ell_c(\kappa; y_o, y_m) = \log g_c(y_o, y_m; \kappa)$$

We cannot estimate the variance parameters $\kappa$ using the usual maximum likelihood procedure as we have incomplete data. The EM algorithm approaches the problem of estimating the variance parameters using the simple complete data form because it is easy to manipulate and easy to use in applications. Before we define the EM algorithm and how it deals with incomplete data, first let us look at the motivation behind it.
Meng and Rubin (1994) state that the idea behind the EM algorithm comes from an old ad hoc idea for handling missing data: (i) if the missing data values were known, then the unknown parameters could be estimated using complete data techniques, and (ii) if the parameters were known, then the missing values could be imputed. Hence, a way forward it to iterate between (i) and (ii) until no changes occur in the parameter estimates and the imputed values. The EM algorithm makes use of the fact that maximum likelihood estimates depend on the data only through sufficient statistics. Hence, rather than imputing the missing values, the EM algorithm imputes that complete data sufficient statistics.

The density function of the observed data can be written as
\[ g(y_o; \kappa) = \frac{g_c(y_o, y_m; \kappa)}{f(y_o, y_m|y_o; \kappa)} \]
where the conditional density of the complete data given the observed data is denoted by \( f(y_o, y_m|y_o; \kappa) \) and is equivalent to \( f(y_m|y_o; \kappa) \). The log-likelihood function for the observed data can now be expressed as
\[ \ell(\kappa; y_o) = \log g(y_o; \kappa) = \log g_c(y_o, y_m; \kappa) - \log f(y_o, y_m|y_o; \kappa) \] (4.2)

Suppose that the density \( g_c(y_o, y_m; \kappa) \) is from an exponential family, that is
\[ g_c(y_o, y_m; \kappa) = b(y_o, y_m) \exp(\kappa^t(y_o, y_m)) \]
where \( t(y_o, y_m) \) is a complete data sufficient statistic and \( b(y_o, y_m) \) and \( a(\kappa) \) are scalar functions where \( a(\kappa) \) is given by
\[ a(\kappa) = \int \int b(y_o, y_m) \exp(\kappa^t(y_o, y_m)) \, dy_o \, dy_m \]

Similarly suppose the conditional density of the complete data given the observed data is given by
\[ f(y_o, y_m|y_o; \kappa) = \frac{b(y_o, y_m) \exp(\kappa^t(y_o, y_m))}{a(\kappa|y_o)} \]

where
\[ a(\kappa|y_o) = \int b(y_o, y_m) \exp(\kappa^t(y_o, y_m)) \, dy_m \]

We can now write the log-likelihood function for the observed data in the following way
\[ \ell(\kappa; y_o) = (\log b(y_o, y_m) + \kappa^t(y_o, y_m) - \log a(\kappa)) - (\log b(y_o, y_m) + \kappa^t(y_o, y_m) - \log a(\kappa|y_o)) \]
\[ = \log a(\kappa|y_o) - \log a(\kappa) \]

To obtain estimates of \( \kappa \) we need to maximise the log-likelihood function with respect to \( \kappa \). One approach is to differentiate the log-likelihood function with respect to \( \kappa \) and equate to zero.
\[ \frac{\partial}{\partial \kappa} \ell(\kappa; y_o) = \frac{\partial}{\partial \kappa} \log a(\kappa|y_o) - \frac{\partial}{\partial \kappa} \log a(\kappa) = 0 \]
Now
\[
\frac{\partial}{\partial \kappa} \log a(\kappa) = \frac{1}{a(\kappa)} \frac{\partial}{\partial \kappa} a(\kappa)
\]
\[
= \frac{1}{a(\kappa)} \int \int b(y_o, y_m) \frac{\partial}{\partial \kappa} \exp(\kappa' t(y_o, y_m)) dy_o dy_m
\]
\[
= \frac{1}{a(\kappa)} \int \int b(y_o, y_m) \exp(\kappa' t(y_o, y_m)) dy_o dy_m
\]
\[
= \int \int b(y_o, y_m) \exp(\kappa' t(y_o, y_m)) a(\kappa) dy_o dy_m
\]
\[
= \int \int t(y_o, y_m) g_c(y_o, y_m; \kappa) dy_o dy_m = E[t(y_o, y_m); \kappa]
\]

In a similar fashion
\[
\frac{\partial}{\partial \kappa} \log a(\kappa|y_o) = E[t(y_o, y_m)|y_o; \kappa]
\]

Therefore the derivative of the log-likelihood function for the observed data with respect to \( \kappa \) can be written as
\[
\frac{\partial}{\partial \kappa} \ell(\kappa; y_o) = E[t(y_o, y_m)|y_o; \kappa] - E[t(y_o, y_m); \kappa]
\]

Equating to zero gives
\[
E[t(y_o, y_m)|y_o; \kappa] = E[t(y_o, y_m); \kappa]
\]

Thus estimation of \( \kappa \) involves equating the expected value of the complete data sufficient statistic and the conditional expectation of the sufficient statistic given the observed data. This leads naturally to a two-stage estimation process.

Given an initial estimate of \( \kappa = \kappa^{(0)} \), the first stage is to calculate the conditional expectation of the sufficient statistic given the observed data as an estimate of the sufficient statistic,
\[
E[t(y_o, y_m)|y_o; \kappa^{(0)}] = t^{(0)}
\]

This is called the Expectation step or the E step. The second stage is then to obtain an update of \( \kappa = \kappa^{(1)} \), as a solution to the equations
\[
t^{(0)} = E[t(y_o, y_m); \kappa]
\]

This is called the Maximisation step or the M step. The combination of the E and the M step completes the first iteration. The E step is then repeated using \( \kappa^{(1)} \) and then the M step allows us to find \( \kappa^{(2)} \). This is the second iteration. On the \((m + 1)\)th iteration, the E step is to calculate
\[
E[t(y_o, y_m)|y_o; \kappa^{(m)}] = t^{(m)}
\] (4.3)
and the M step is to obtain $\kappa^{(m+1)}$ by solving
\[
t^{(m)} = E[t(y_o, y_m); \kappa] \tag{4.4}
\]
If the algorithm converges to $\kappa = \hat{\kappa}$ then combining (4.3) and (4.4) gives
\[
E[t(y_o, y_m)|y_o; \hat{\kappa}] = E[t(y_o, y_m); \hat{\kappa}]
\]
or equivalently
\[
\frac{\partial}{\partial \kappa} \ell(\kappa; y_o) = 0
\]
at $\kappa = \hat{\kappa}$. Therefore we can see that repeated application of the E and M steps leads to the value $\hat{\kappa}$ that maximises the observed data likelihood $\ell(\kappa, y_o)$.

The use of the EM algorithm extends beyond the exponential family. An alternative approach which generalises beyond the exponential family is to consider the log-likelihood function of the complete data rather than the observed data. If the density function of the complete data is of the form of an exponential family, the log-likelihood function of the complete data is given by
\[
\ell_c(\kappa; y_o, y_m) = \log g_c(y_o, y_m; \kappa) = -\log a(\kappa) + \log b(y_o, y_m) + \kappa^t(y_o, y_m)
\]
In the previous approach the E step involved taking the conditional expectation of the complete data sufficient statistic given the observed data at $\kappa = \kappa^{(m)}$. In this more general approach we will take the conditional expectation of the complete data log-likelihood function given the observed data at a specified $\kappa = \kappa^{(m)}$. Thus
\[
E[\ell_c(\kappa; y_o, y_m)|y_o; \kappa^{(m)}] = -\log a(\kappa) + E[\log b(y_o, y_m)|y_o; \kappa^{(m)}] + \kappa^t(y_o, y_m)
\]
constitutes the E step. The M step then involves maximising this conditional expectation with respect to $\kappa$ to obtain an updated estimate of $\kappa$. Differentiating (4.5) we have
\[
\frac{\partial}{\partial \kappa} E[\ell_c(\kappa; y_o, y_m)|y_o; \kappa^{(m)}] = -\frac{\partial}{\partial \kappa} \log a(\kappa) + E[t(y_o, y_m)|y_o; \kappa^{(m)}] = -E[t(y_o, y_m); \kappa] + E[t(y_o, y_m)|y_o; \kappa^{(m)}]
\]
and equating to zero gives
\[
E[t(y_o, y_m)|y_o; \kappa^{(m)}] = E[t(y_o, y_m); \kappa]
\]
Hence we can see that the second approach of considering the conditional expectation of the complete data log-likelihood function given the observed data at $\kappa = \kappa^{(m)}$, leads to the same equations as the first approach of considering the log-likelihood of the observed data. So there are two equivalent approaches to estimating the variance parameters when the densities belong to the exponential family.
4.2 Definition of the EM Algorithm

In the general case, when the densities are not assumed to be from exponential families, the second approach is adopted. The idea behind the second approach being that the observed data log-likelihood is solved indirectly by proceeding in terms of the complete data log-likelihood. The complete data is unobservable and so its log-likelihood is replaced by its expectation conditional on the observed data.

Dempster et al. (1977) define the E step of the EM algorithm as the calculation of a function $Q(\kappa; \kappa^{(m)})$ at the current $\kappa = \kappa^{(m)}$ such that

$$Q(\kappa; \kappa^{(m)}) = \mathbb{E}[\ell_c(\kappa; y_o, y_m) \mid y_o; \kappa^{(m)}] = \int \ell_c(\kappa; y_o, y_m) f(y_o, y_m \mid y_o; \kappa^{(m)}) \, dy_m$$

The M step is then to choose $\kappa^{(m+1)}$ to maximise $Q(\kappa; \kappa^{(m)})$, so that

$$Q(\kappa^{(m+1)}; \kappa^{(m)}) \geq Q(\kappa; \kappa^{(m)})$$

McLachlan and Krishnan (1997) suggest that the E and M steps are repeated until the difference

$$\ell(\kappa^{(m+1)}; y_o, y_m) - \ell(\kappa^{(m)}; y_o, y_m)$$

changes by an arbitrarily small amount. Foulley et al. (2000) propose that the iterations stop when the norm of $\kappa = (\gamma', \phi')'$, given by

$$\sqrt{\sum_i \Delta \kappa_i^2 / \sum_i \kappa_i^2}$$

is smaller than $10^{-6}$, for both $\gamma$ and $\phi$, where $\Delta \kappa_i^2$ is the increment in $\kappa_i^2$ from the $(i-1)$th iteration to the $i$th iteration.

4.3 Monotonicity of the EM Algorithm

Dempster et al. (1977) show that the log-likelihood function for the observed data does not decrease after an EM iteration, that is

$$\ell(\kappa^{(m+1)}; y_o) \geq \ell(\kappa^{(m)}; y_o)$$

Here we will present a proof along the lines of McLachlan and Krishnan (1997). From (4.2) we have

$$\ell(\kappa; y_o) = \ell_c(\kappa; y_o, y_m) - \log f(y_o, y_m \mid y_o; \kappa)$$

Taking expectations of both sides conditional on the complete data, evaluating at the current fit for $\kappa$, $\kappa^{(m)}$, gives

$$\mathbb{E}[\ell(\kappa; y_o) \mid y_o; \kappa^{(m)}] = \mathbb{E}[\ell_c(\kappa; y_o, y_m) \mid y_o; \kappa^{(m)}] - \mathbb{E}[\log f(y_o, y_m \mid y_o; \kappa) \mid y_o; \kappa^{(m)}]$$
This can be written as
\[
\ell(\kappa; y_o) = Q(\kappa; \kappa^{(m)}) - H(\kappa; \kappa^{(m)})
\] (4.6)
where \( H(\kappa; \kappa^{(m)}) = \mathbb{E}[\log f(y_o, y_m | y_o; \kappa)|y_o; \kappa^{(m)}] \).

To show that the likelihood for the observed data is not decreased after an iteration of
the EM algorithm we must show that
\[
\ell(\kappa^{(m+1)}; y_o) - \ell(\kappa^{(m)}; y_o) \geq 0
\]
Hence consider
\[
\ell(\kappa^{(m+1)}; y_o) - \ell(\kappa^{(m)}; y_o)
= [Q(\kappa^{(m+1)}; \kappa^{(m)}) - Q(\kappa^{(m)}; \kappa^{(m)})] - [H(\kappa^{(m+1)}; \kappa^{(m)}) - H(\kappa^{(m)}; \kappa^{(m)})]
\]
By definition
\[
Q(\kappa^{(m+1)}; \kappa^{(m)}) \geq Q(\kappa; \kappa^{(m)})
\]
for all \( \kappa \) and therefore
\[
Q(\kappa^{(m+1)}; \kappa^{(m)}) \geq Q(\kappa^{(m)}; \kappa^{(m)})
\]
Hence to show that
\[
\ell(\kappa^{(m+1)}; y_o) - \ell(\kappa^{(m)}; y_o) \geq 0
\]
we need to show that
\[
H(\kappa^{(m+1)}; \kappa^{(m)}) - H(\kappa^{(m)}; \kappa^{(m)}) \leq 0
\]
So we consider
\[
H(\kappa; \kappa^{(m)}) - H(\kappa^{(m)}; \kappa^{(m)})
= \mathbb{E}[\log f(y_o, y_m | y_o; \kappa)|y_o; \kappa^{(m)}] - \mathbb{E}[\log f(y_o, y_m | y_o; \kappa^{(m)})|y_o; \kappa^{(m)}]
= \mathbb{E}[\log \left( \frac{f(y_o, y_m | y_o; \kappa)}{f(y_o, y_m | y_o; \kappa^{(m)})} \right)|y_o; \kappa^{(m)}]
\leq \log E \left[ \frac{f(y_o, y_m | y_o; \kappa)}{f(y_o, y_m | y_o; \kappa^{(m)})} \right]|y_o; \kappa^{(m)}] \quad \text{(using Jensen’s inequality)}
\]
\[
= \log \int \frac{f(y_o, y_m | y_o; \kappa)}{f(y_o, y_m | y_o; \kappa^{(m)})} \cdot f(y_o, y_m | y_o; \kappa^{(m)}) \, dy_m
= \log \int f(y_o, y_m | y_o; \kappa) \, dy_m
= \log 1
= 0
\]
Hence
\[
H(\kappa; \kappa^{(m)}) - H(\kappa^{(m)}; \kappa^{(m)}) \leq 0
\]
for all \( \kappa \) and so
\[
H(\kappa^{(m+1)}; \kappa^{(m)}) - H(\kappa^{(m)}; \kappa^{(m)}) \leq 0
\]
Therefore
\[
\ell(\kappa^{(m+1)}; y_o) - \ell(\kappa^{(m)}; y_o) \geq 0
\]
and we have shown that the log-likelihood of the observed data is not decreased after an
iteration of the EM algorithm and hence the EM algorithm is monotone.
4.4 Self-Consistency of the EM Algorithm

The EM algorithm provides a method of solving the score equations. Hence, at convergence, we would expect that differentiation of the conditional expectation of the log-likelihood of the complete data given the observed data, as calculated in the final E step, would give the score equation for the observed data.

Let the score for the observed data be given by $S(y_o; \kappa)$ and the score for the complete data be given by $S_c(y_o, y_m; \kappa)$. Consider

$$S(y_o; \kappa) = \frac{\partial \ell(\kappa; y_o)}{\partial \kappa} = \frac{\partial \log g(y_o; \kappa)}{\partial \kappa} = \frac{g'(y_o; \kappa)}{g(y_o; \kappa)}$$

Now using (4.1) we have

$$g'(y_o; \kappa) = \int g'(c(y_o, y_m; \kappa)) \, dy_m$$

Hence

$$S(y_o; \kappa) = \int \frac{g'(c(y_o, y_m; \kappa))}{g(y_o; \kappa)} \, dy_m$$

and noting that

$$g'(c(y_o, y_m; \kappa)) = \frac{\partial \log g_c(y_o, y_m; \kappa)}{\partial \kappa} g_c(y_o, y_m; \kappa)$$

we have

$$S(y_o; \kappa) = \int \frac{\partial \log g_c(y_o, y_m; \kappa)}{\partial \kappa} g_c(y_o, y_m; \kappa) \, dy_m$$

$$= \int \frac{\partial c(y_o, y_m; \kappa)}{\partial \kappa} g(y_m | y_o) \, dy_m$$

$$= \int S_c(y_o, y_m; \kappa) g(y_m | y_o) \, dy_m$$

$$= E[S_c(y_o, y_m; \kappa) | y_o]$$

Hence the score for the observed data $S(y_o; \kappa)$ can be expressed as the conditional expectation of the score for the complete data $S_c(y_o, y_m; \kappa)$ given the observed data $y_o$.

We can now write the score for the observed data as

$$S(y_o; \kappa) = E\left[ \frac{\partial}{\partial \kappa} \ell_c(\kappa; y_o, y_m) | y_o \right]$$

$$= \frac{\partial}{\partial \kappa} E[\ell_c(\kappa; y_o, y_m) | y_o]$$

Hence we have shown that the score for the observed data can be written as the derivative of the conditional expectation of the log-likelihood of the complete data given the observed data, proving that the EM algorithm is self-consistent.
4.5 REML EM Algorithm for Mixed Models

In this section we develop the REML EM algorithm for obtaining estimates of the variance parameters $\kappa$ for the general linear mixed model given in (2.1).

Recall that each iteration of the EM algorithm consists of two steps; the E step and the M step. The E step involves calculating the conditional expectation of the complete data log-likelihood function given the observed data at the current fit of the variance parameters. The M step then involves maximisation of this conditional expectation with respect to variance parameters.

In the case of linear mixed models, the random effects $u$ are thought of as the missing data. Hence the complete data $y_c$ is made up of the observed data $y$ and the unobserved random effects $u$. The distribution of the complete data is given by

$$
\begin{bmatrix}
  y \\
  u
\end{bmatrix} \sim N \left( \begin{bmatrix} X\tau \\
  0 
\end{bmatrix}, \theta \begin{bmatrix}
  ZGZ' + R & ZG \\
  GZ' & G
\end{bmatrix} \right)
$$

The joint log-likelihood function of the complete data can be written as the sum of the log-likelihood of the missing data $u$ and the conditional log-likelihood of the observed data $y$ given the missing data $u$, namely

$$
\ell_c(\theta, \kappa; y, u) = \ell_c(\theta, \kappa; y) + \ell_c(\theta, \kappa; y|u)
$$

Note that $\ell_c(\theta, \kappa; y|u)$ is not strictly a log-likelihood since $u$ is not observed.

Given that we have an update for $\theta = \theta^{(m)}$, given in (2.54), we evaluate the complete data log-likelihood at this value. The complete data log-likelihood evaluated at $\theta = \theta^{(m)}$ is given by

$$
\ell_c(\kappa) = \ell_c(\theta, \kappa; y, u, \theta^{(m)})
= \ell_c(\theta, \kappa; u, \theta^{(m)}) + \ell_c(\theta, \kappa; y|u, \theta^{(m)})
$$

It is this form of the complete data log-likelihood that we use in the EM algorithm to obtain an update for $\kappa$ and it will be denoted by $\ell_c(\kappa)$.

The log-likelihood of the missing data $u$ evaluated at $\theta = \theta^{(m)}$, excluding constant terms, is given by

$$
\ell_c(\gamma) = \ell_c(\theta, \kappa; u, \theta^{(m)})
= -\frac{1}{2} \left(b \log \theta^{(m)} + \log |G| + \frac{u'G^{-1}u}{\theta^{(m)}} \right)
$$

This is a function of $\gamma$ only and will be denoted by $\ell_c(\gamma)$.

Using Result A.11, the conditional distribution of the observed data $y$ given the missing data $u$ is given by

$$
y|u \sim N(X\tau + Zu, \theta R)
$$
where $R = \sigma^2 \Sigma$. Hence the log-likelihood of $y|u$ evaluated at $\theta = \theta^{(m)}$, excluding any constant terms, is given by

$$
\ell_c(\sigma^2, \phi) = \ell_c(\theta, \kappa; y|u, \theta^{(m)}) = -\frac{1}{2} \left( n \log \theta^{(m)} + n \log \sigma^2 + \log |\Sigma| + \frac{(y - X\tau - Zu)'\Sigma^{-1}(y - X\tau - Zu)}{\sigma^2 \theta^{(m)}} \right)
$$

$$
= -\frac{1}{2} \left( n \log \theta^{(m)} + n \log \sigma^2 + \log |\Sigma| + \frac{e'\Sigma^{-1}e}{\sigma^2 \theta^{(m)}} \right)
$$

(4.10)

This is a function of $\sigma^2$ and $\phi$ only and it will be denoted by $\ell_c(\sigma^2, \phi)$.

Combining (4.9) and (4.10) allows the complete data log-likelihood in (4.8)

$$
\ell_c(\kappa) = \ell_c(\gamma) + \ell_c(\sigma^2, \phi)
$$

$$
= -\frac{1}{2} \left( b \log \theta^{(m)} + \log |G| + \frac{u'G^{-1}u}{\theta^{(m)}} \right) - \frac{1}{2} \left( n \log \theta^{(m)} + n \log \sigma^2 + \log |\Sigma| + \frac{e'\Sigma^{-1}e}{\sigma^2 \theta^{(m)}} \right)
$$

(4.11)

The E step involves calculating the conditional expectation of the complete data log-likelihood in (4.11) given the observed data at the current fit of the variance parameters, that is $\theta = \theta^{(m)}$ and $\kappa = \kappa^{(m)}$. For ordinary maximum likelihood estimation of variance parameters we would condition on $y$. However, as we are looking at REML estimation, we condition on $y_2$, as defined in Section 2.2.2. So the E step for the EM algorithm for REML estimation for mixed models is given by

$$
E[\ell_c(\kappa)|y_2; \theta^{(m)}, \kappa^{(m)}] = E[\ell_c(\gamma)|y_2; \theta^{(m)}, \kappa^{(m)}] + E[\ell_c(\sigma^2, \phi)|y_2; \theta^{(m)}, \kappa^{(m)}]
$$

For simplicity we will use the operand $E_{EM}[\cdot]^{(m)}$ to denote the conditional expectation of a function, given $y_2$ at the current fit for $\theta = \theta^{(m)}$ and $\kappa = \kappa^{(m)}$. Hence the E step can be written as

$$
E_{EM}[\ell_c(\kappa)]^{(m)} = E_{EM}[\ell_c(\gamma)]^{(m)} + E_{EM}[\ell_c(\sigma^2, \phi)]^{(m)}
$$

(4.12)

Note that the E step consists of two parts; one is a function of $\gamma$ and the other is a function of $\sigma^2$ and $\phi$. We refer to these two parts as the E step for $\gamma$ and the E step for $\sigma^2$ and $\phi$.

First we will calculate the E step for $\gamma$, that is the conditional expectation of the log-likelihood of $u$ as given in (4.9), given $y_2$ at the current fit for $\theta = \theta^{(m)}$ and $\kappa = \kappa^{(m)}$.

$$
E_{EM}[\ell_c(\gamma)]^{(m)} = E[\ell_c(\gamma)|y_2; \theta^{(m)}, \kappa^{(m)}] = -\frac{1}{2} \left( b \log \theta^{(m)} + \log |G| + \frac{E_{EM}[u'G^{-1}u]^{(m)}}{\theta^{(m)}} \right)
$$

(4.13)

To calculate $E_{EM}[u'G^{-1}u]^{(m)} = E[u'G^{-1}u|y_2; \theta^{(m)}, \kappa^{(m)}]$ we need the distribution of $u$ conditional on $y_2$. The joint distribution of $u$ and $y_2$ is given by

$$
\begin{bmatrix}
  u \\
  y_2
\end{bmatrix} \sim N \left( \begin{bmatrix} 0 \\
  0 \end{bmatrix}, \theta \begin{bmatrix} G & GZ' L_2 \\
  L_2' Z G & L_2' H L_2 \end{bmatrix} \right)
$$
Using Result A.11, the conditional distribution of \( u \) given \( y_2 \) is
\[
u | y_2 \sim N(GZ'Py, \theta(G - GZ'PZG))
\sim N(\tilde{u}, \theta C^{ZZ})
\]
using (2.7) and (2.20). Therefore using Result A.10
\[
E_{EM}[\nu'G^{-1}\nu]^{(m)} = \theta^{(m)} \text{tr}(C^{ZZ(m)}G^{-1}) + \hat{u}^{(m)}'G^{-1}\hat{u}^{(m)}
\]
We can now write the E step for \( \gamma \) in (4.13) as
\[
E_{EM}[\ell_c(\gamma)]^{(m)} = -\frac{1}{2} \left( b \log \theta^{(m)} + \log |G| + \text{tr}(C^{ZZ(m)}G^{-1}) + \frac{\hat{u}^{(m)}'G^{-1}\hat{u}^{(m)}}{\theta^{(m)}} \right)
\]
(4.14)

Now we will calculate the E step for \( \sigma^2 \) and \( \phi \), that is the conditional expectation of the log-likelihood of \( y \) given \( u \) conditional on \( y_2 \) at the current fit for \( \theta = \theta^{(m)} \) and \( \kappa = \kappa^{(m)} \).
\[
E_{EM}[\ell_c(\sigma^2, \phi)]^{(m)} = E[\ell_c(\sigma^2, \phi)|y_2; \theta^{(m)}, \kappa^{(m)}]
\]
\[
= -\frac{1}{2} \left( n \log \theta^{(m)} + n \log \sigma^2 + \log |\Sigma| + \frac{E_{EM}[e'\Sigma^{-1}e]^{(m)}}{\sigma^2\theta^{(m)}} \right)
\]
(4.15)

To calculate \( E_{EM}[e'\Sigma^{-1}e]^{(m)} = E[e'\Sigma^{-1}e|y_2; \theta^{(m)}, \kappa^{(m)}] \) we need the distribution of \( e \) conditional on \( y_2 \). The joint distribution of \( e \) and \( y_2 \) is given by
\[
\begin{bmatrix} e \\ y_2 \end{bmatrix} \sim N \left( \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \theta \begin{bmatrix} R & RL_2 \\ L_2R & L_2HL_2 \end{bmatrix} \right)
\]
Using Result A.11, the conditional distribution of \( e \) given \( y_2 \) is given by
\[
e | y_2 \sim N(RPy, \theta(R - RPR))
\sim N(\tilde{e}, \theta(R - RPR))
\]
(4.16)

using (2.16). Noting that
\[
R - RPR = R - R(R^{-1} - R^{-1}WC^{-1}W'R^{-1})R
= WC^{-1}W'
\]
the conditional distribution of \( e \) given \( y_2 \) is
\[
e | y_2 \sim N(\tilde{e}, \theta WC^{-1}W')
\]
(4.17)

Therefore using Result A.10
\[
E_{EM}[e'\Sigma^{-1}e]^{(m)} = \theta^{(m)} \text{tr}(WC^{-1}(m)W'S^{-1}) + \tilde{e}^{(m)}'\Sigma^{-1}\tilde{e}^{(m)}
\]
We can now write the E step for \( \sigma^2 \) and \( \phi \) in (4.15) as
\[
E_{EM}[\ell_c(\sigma^2, \phi)]^{(m)}
\]
\[
= -\frac{1}{2} \left( n \log \theta^{(m)} + n \log \sigma^2 + \log |\Sigma| + \frac{\text{tr}(WC^{-1}(m)W'S^{-1})}{\sigma^2} + \frac{\tilde{e}^{(m)}'\Sigma^{-1}\tilde{e}^{(m)}}{\sigma^2\theta^{(m)}} \right)
\]
(4.18)
Combining (4.14) and (4.18), the E step as given in (4.12) can be written as

\[
E_{EM}[\ell_c(\kappa)]^{(m)} = E_{EM}[\ell_c(\gamma)]^{(m)} + E_{EM}[\ell_c(\sigma^2, \phi)]^{(m)}
\]

\[
= -\frac{1}{2} \left( b \log \theta^{(m)} + \log |G| + \text{tr}(C^{ZZ(m)}G^{-1}) + \frac{\tilde{u}^{(m)}'G^{-1}\tilde{u}^{(m)}}{\theta^{(m)}} \right)
\]

\[
- \frac{1}{2} \left( n \log \theta^{(m)} + n \log \sigma^2 + \log |\Sigma| + \frac{\text{tr}(WC^{-1}(m)W'\Sigma^{-1})}{\sigma^2} + \frac{\tilde{e}^{(m)}'\Sigma^{-1}\tilde{e}^{(m)}}{\sigma^2\theta^{(m)}} \right)
\]

(4.19)

The M step requires maximisation of the conditional expectation calculated in the E step in (4.19) with respect to the variance parameters \(\gamma\), \(\sigma^2\) and \(\phi\), to obtain updates of these variance parameters. The E step in (4.12) is separated into 2 parts, the E step for \(\gamma\) as given in (4.14), and the E step for \(\sigma^2\) and \(\phi\) as given in (4.18). So when we differentiate with respect to the variance parameters in the M step, we consider the 2 parts separately; the E step for \(\gamma\) is differentiated with respect to \(\gamma\) and the E step for \(\sigma^2\) and \(\phi\) is differentiated with respect to \(\sigma^2\) and \(\phi\).

The derivative associated with the M step for \(\gamma_{ij}\) is given by

\[
\frac{\partial E_{EM}[\ell_c(\gamma)]^{(m)}}{\partial \gamma_{ij}} = \frac{\partial E_{EM}[\ell_c(\gamma)]^{(m)}}{\partial \gamma_{ij}}
\]

\[
= -\frac{1}{2} \left( \frac{\partial \log |G|}{\partial \gamma_{ij}} + \frac{\text{tr}(C^{ZZ(m)}G^{-1})}{\partial \gamma_{ij}} + \frac{1}{\theta^{(m)}} \frac{\partial \tilde{u}^{(m)'G^{-1}\tilde{u}^{(m)}}}{\partial \gamma_{ij}} \right)
\]

(4.20)

From (2.2) we have

\[
\log |G| = \log |G_1| + \ldots + \log |G_i| + \ldots + \log |G_q|
\]

Using Result A.1, note that

\[
\frac{\partial \log |G_i|}{\partial \gamma_{ij}} = \text{tr}(G_i^{-1}G_{ij})
\]

and hence the first term in (4.20) is given by

\[
\frac{\partial \log |G|}{\partial \gamma_{ij}} = \frac{\partial \log |G_i|}{\partial \gamma_{ij}}
\]

\[
= \text{tr}(G_i^{-1}G_{ij})
\]

Now consider

\[
\text{tr}(C^{ZZ}G^{-1}) = \text{tr}(C^{Z_1Z_1}G_1^{-1}) + \ldots + \text{tr}(C^{Z_iZ_i}G_i^{-1}) + \ldots + \text{tr}(C^{Z_qZ_q}G_q^{-1})
\]

Using Result A.1, note that

\[
\frac{\partial G_i^{-1}}{\partial \gamma_{ij}} = -G_i^{-1}G_{ij}G_i^{-1}
\]
and hence the second term in (4.20) is given by
\[
\frac{\partial}{\partial \gamma_{ij}} \text{tr}(CZ(m)Z(m)G^{-1}) = \frac{\partial}{\partial \gamma_{ij}} \text{tr}(CZ_iZ_i(m)G_i^{-1})
\]
\[
= \text{tr} \left( CZ_iZ_i(m) \frac{\partial G_i^{-1}}{\partial \gamma_{ij}} \right)
\]
\[
= -\text{tr}(CZ_iZ_i(m)G_i^{-1}G_i^{-1}G_i^{-1})
\]
Finally consider
\[
\tilde{u}'G^{-1}\tilde{u} = \tilde{u}_i'G_i^{-1}\tilde{u}_i + \ldots + \tilde{u}_i'G_i^{-1}\tilde{u}_i + \ldots + \tilde{u}_q'G_q^{-1}\tilde{u}_q
\]
and hence the third term in (4.20) is given by
\[
\frac{\partial}{\partial \gamma_{ij}} \tilde{u}(m)'G^{-1}\tilde{u}(m) = \frac{\partial}{\partial \gamma_{ij}} \tilde{u}(m)_i'G_i^{-1}\tilde{u}(m)_i
\]
\[
= \tilde{u}(m)_i'G_i^{-1}G_i^{-1}\tilde{u}(m)_i
\]
This allows the derivative associated with the M step for \(\gamma_{ij}\) as given in (4.20) to be written as
\[
\frac{\partial E_{EM}[\ell_c(\kappa)](m)}{\partial \gamma_{ij}} = -\frac{1}{2} \left( \text{tr}(G_i^{-1}\dot{G}_i) - \text{tr}(CZ_iZ_i(m)G_i^{-1}\dot{G}_iG_i^{-1}) - \tilde{u}(m)_i'G_i^{-1}\dot{G}_iG_i^{-1}\tilde{u}(m)_i \right)
\]
(4.21)
To obtain an update, \(\gamma_{ij}^{(m+1)}\), we equate (4.21) to zero and solve for \(\gamma_{ij}\). This will be illustrated by way of example in the following chapters.

It was shown in Section 4.4 that for the general case, the derivative of the conditional expectation of the complete data log-likelihood given the observed data as calculated in the final E step, is consistent with the score equation for the observed data. At convergence, the derivative associated with the M-step for \(\gamma_{ij}\) is given by
\[
\frac{\partial E_{EM}[\ell_c(\kappa)]}{\partial \gamma_{ij}} = -\frac{1}{2} \left( \text{tr}(G_i^{-1}\dot{G}_i) - \text{tr}(CZ_iZ_i(m)G_i^{-1}\dot{G}_iG_i^{-1}) - \tilde{u}(m)_i'G_i^{-1}\dot{G}_iG_i^{-1}\tilde{u}(m)_i \right)
\]
which is equivalent to the REML score for \(\gamma_{ij}\) given in (2.44).

In the case when there is only one random term, the derivative associated with the M step for \(\gamma_i\) is given by
\[
\frac{\partial E_{EM}[\ell_c(\kappa)](m)}{\partial \gamma_i} = -\frac{1}{2} \left( \text{tr}(G^{-1}\dot{G}_i) - \text{tr}(CZZ(m)G^{-1}\dot{G}_iG^{-1}) - \tilde{u}(m)'G^{-1}\dot{G}_iG^{-1}\tilde{u}(m) \right)
\]
(4.22)
We obtain an update for $\gamma_i$ by equating (4.22) to zero and solving for $\gamma_i$. Again this will be illustrated by way of example in the following chapters. Note that at convergence (4.22) is equivalent to the REML score for $\gamma_i$ in (2.45).

The derivative associated with the M step for $\sigma^2$ is given by

$$
\frac{\partial E_{EM}[\ell_c(\kappa)]^{(m)}}{\partial \sigma^2} = \frac{\partial E_{EM}[\ell_c(\sigma^2, \phi)]^{(m)}}{\partial \sigma^2} = -\frac{1}{2} \left( \frac{n}{\sigma^2} - \frac{\text{tr}(WC^{-1(m)}W'\Sigma^{-1})}{\sigma^4} - \frac{\bar{e}^{(m)'\Sigma^{-1}e^{(m)}}}{\sigma^4\theta^{(m)}} \right) \tag{4.23}
$$

The EM update for $\sigma^2$ is obtained by equating (4.23) to zero and solving for $\sigma^2$.

$$
\sigma^{2(m+1)} = \frac{1}{n} \left( \text{tr}(WC^{-1(m)}W'\Sigma^{-1}) + \frac{\bar{e}^{(m)'\Sigma^{-1}e^{(m)}}}{\theta^{(m)}} \right) \tag{4.24}
$$

Note that at convergence, the derivative associated with the M-step for $\sigma^2$ is given by

$$
\frac{\partial E_{EM}[\ell_c(\kappa)]}{\partial \sigma^2} = -\frac{1}{2} \left( \frac{n}{\sigma^2} - \frac{\text{tr}(WC^{-1}W'\Sigma^{-1})}{\sigma^4} - \frac{\bar{e}\Sigma^{-1}\bar{e}}{\sigma^4\theta} \right)
$$

Therefore at convergence, the derivative associated with the M-step for $\sigma^2$ is equivalent to the REML score for $\sigma^2$ in (2.48)

The derivative associated with the M step for $\phi_i$ is given by

$$
\frac{\partial E_{EM}[\ell_c(\kappa)]^{(m)}}{\partial \phi_i} = \frac{\partial E_{EM}[\ell_c(\sigma^2, \phi)]^{(m)}}{\partial \phi_i} = -\frac{1}{2} \left( \frac{\partial \log |\Sigma|}{\partial \phi_i} + \frac{1}{\sigma^2} \frac{\partial \text{tr}(WC^{-1(m)}W'\Sigma^{-1})}{\partial \phi_i} + \frac{1}{\sigma^2\theta^{(m)}} \frac{\partial \bar{e}^{(m)'\Sigma^{-1}e^{(m)}}}{\partial \phi_i} \right) \tag{4.25}
$$

Using Result A.1, note that

$$
\frac{\partial \log |\Sigma|}{\partial \phi_i} = \text{tr}(\Sigma^{-1}\dot{\Sigma}_i) \tag{4.27}
$$

and

$$
\frac{\partial \Sigma^{-1}}{\partial \phi_i} = -\Sigma^{-1}\dot{\Sigma}_i\Sigma^{-1}
$$

Hence the second term in (4.26) is given by

$$
\frac{\partial}{\partial \phi_i} \text{tr}(WC^{-1(m)}W'\Sigma^{-1}) = \text{tr} \left( WC^{-1(m)}W' \frac{\partial \Sigma^{-1}}{\partial \phi_i} \right) = -\text{tr}(WC^{-1(m)}W'\Sigma^{-1}\dot{\Sigma}_i\Sigma^{-1}) \tag{4.28}
$$
and the third term in (4.26) is given by
\[
\frac{\partial}{\partial \phi_i} \hat{e}^{(m)'} \Sigma^{-1} \hat{e}^{(m)} = \hat{e}^{(m)'} \frac{\partial \Sigma^{-1}}{\partial \phi_i} \hat{e}^{(m)}
\]
\[
= -\hat{e}^{(m)'} \Sigma^{-1} \dot{\Sigma}_i \Sigma^{-1} \hat{e}^{(m)}
\] (4.29)

This allows the derivative associated with the M step for \( \phi_i \) as given in (4.26) to be written as
\[
\frac{\partial E_{EM}[\ell_c(\kappa)]^{(m)}}{\partial \phi_i} = -\frac{1}{2} \left( \text{tr}(\Sigma^{-1} \dot{\Sigma}_i) - \frac{\text{tr}(WC^{-1(m)}W' \Sigma^{-1} \dot{\Sigma}_i \Sigma^{-1})}{\sigma^2} - \frac{\hat{e}^{(m)'} \Sigma^{-1} \dot{\Sigma}_i \Sigma^{-1} \hat{e}^{(m)}}{\sigma^2 \theta^{(m)}} \right)
\] (4.30)

To obtain an update, \( \phi_i^{(m+1)} \), we equate (4.30) to zero and solve for \( \phi_i \). This will also be illustrated by way of example in the following chapters.

At convergence, the derivative associated with the M-step for \( \phi_i \) is given by
\[
\frac{\partial E_{EM}[\ell_c(\kappa)]}{\partial \phi_i} = -\frac{1}{2} \left( \text{tr}(\Sigma^{-1} \dot{\Sigma}_i) - \frac{\text{tr}(WC^{-1}W' \Sigma^{-1} \dot{\Sigma}_i \Sigma^{-1})}{\sigma^2} - \frac{\hat{e} \Sigma^{-1} \dot{\Sigma}_i \Sigma^{-1} \hat{e}}{\sigma^2 \theta} \right)
\]
which is equivalent to the REML score for \( \phi_i \) given in (2.52).
Chapter 5

PXEM Algorithm

5.1 Introduction

The EM algorithm is often criticised for being slow to converge. To improve the rate of convergence Liu et al. (1998) introduced a new algorithm called the Parameter Expanded EM (PX-EM) algorithm. The parameter space of the complete data model is expanded to a larger set of parameters $\mathcal{K} = (\kappa^*, \lambda)'$, where $\kappa^*$ plays the same role in the larger model as $\kappa$ does in the original model, and $\lambda$ is a working parameter that is fixed at $\lambda_0$ in the original model. Formally $\mathcal{K}$ satisfies 2 conditions:

1. The observed data model is preserved in the sense that $\mathcal{K}$ can be reduced to the original parameter $\kappa$ via a many-to-one reduction function $R$, that is $\kappa = R(\mathcal{K})$.

2. When $\lambda$ is set to its null value, $\lambda_0$, $\mathcal{K}$ induces the same complete data model as with $\kappa = \kappa^*$.

The PX-EM Algorithm, like the EM Algorithm, consists of two steps, an E-step and an M-step. The PX-E step computes the conditional expectation of the complete data log-likelihood given the observed data at $\mathcal{K}^{(m)} = (\kappa^{(m)}', \lambda_0')$. 

\[
Q(\mathcal{K}; \mathcal{K}^{(m)}) = E[\ell_c(\mathcal{K}; y_o, y_m) | y_o; \mathcal{K}^{(m)}]
\]

The PX-M step then maximises this conditional expectation with respect to the expanded parameters to obtain $\mathcal{K}^{(m+1)}$. That is $\mathcal{K}^{(m)}$ is chosen such that 

\[
Q(\mathcal{K}^{(m+1)}; \mathcal{K}^{(m)}) \geq Q(\mathcal{K}; \mathcal{K}^{(m)})
\]

and $\kappa$ is updated via the reduction function $R(\mathcal{K})$ to obtain $\kappa^{(m)} = R(\mathcal{K}^{(m)})$.

Since the observed data model is preserved, the observed data log-likelihood is not altered by the introduction of the working parameter $\lambda$, hence $\ell(\mathcal{K}; y_o) = \ell(\kappa; y_o)$. Therefore 

\[
\ell(\kappa; y_o) = \ell_c(\mathcal{K}; y_o, y_m) - \log f(y_o, y_m | y_o; \mathcal{K})
\]
Taking expectations of both sides conditional on the complete data, evaluating at the current fit for $\mathcal{K}, \mathcal{K}^{(m)}$, gives

$$E[\ell(\kappa; y_o \mid y_o; \mathcal{K})] = E[\ell_c(\mathcal{K}; y_o, y_m) | y_o; \mathcal{K}^{(m)}] - E[\log f(y_o, y_m | y_o; \mathcal{K}) | y_o; \mathcal{K}^{(m)}]$$

This can be written as

$$\ell(\kappa; y_o) = Q(\mathcal{K}; \mathcal{K}^{(m)}) - H(\mathcal{K}; \mathcal{K}^{(m)})$$

where $H(\mathcal{K}; \mathcal{K}^{(m)}) = E[\log f(y_o, y_m | y_o; \mathcal{K}) | y_o; \mathcal{K}^{(m)}]$.

Following the methodology of (4.3), it is easily shown that the PX-EM algorithm maintains the monotone convergence of the EM algorithm.

### 5.2 REML PXEM Algorithm for Mixed Models

Liu et al. (1998) applied the PXEM algorithm to a range of models, including a random coefficient linear mixed model. Foulley and van Dyk (2000) presented the PXEM algorithm for the general linear mixed model. They describe the PXEM algorithm for mixed models as involving a rescaling of the random effects.

The general linear mixed model as given in (2.1) is

$$y = X\tau + Zu + e$$

where $u \sim N(0, \theta G)$, $e \sim N(0, \theta R)$ and $y \sim N(X\tau, \theta H)$, $H = ZGZ' + R$. Recall that $G$ is a function of $\gamma$ and $R = \sigma^2 \Sigma$ where $\Sigma$ is a function of $\phi$.

We expand this model to

$$y = X\tau + Z\Lambda f + e$$

where $u = \Lambda f$, $\Lambda = \Lambda(\lambda)$ is a $b \times b$ invertible matrix, $f$ is a $b \times 1$ vector of rescaled random effects. The joint distribution of $f$ and $e$ is given by

$$\begin{bmatrix} f \\ e \end{bmatrix} \sim N\left( \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \theta \begin{bmatrix} D & 0 \\ 0 & R \end{bmatrix} \right)$$

$D = D(d)$ is a symmetric positive definite variance matrix that can be partitioned in the same way as $G$ in (2.2). The vectors $d$ and $\lambda$ are vectors of variance parameters. Note that

$$\begin{align*}
\text{Var}(u) &= \text{Var}(\Lambda f) \\
\theta G &= \Lambda \text{Var}(f) \Lambda' \\
&= \theta \Lambda D \Lambda'
\end{align*}$$

and hence

$$G = \Lambda D \Lambda'$$
The variance parameters of the expanded model in (5.2) are \( \theta \) and \( \mathbf{K}' = (\kappa', \lambda') \) where \( \kappa' = (\mathbf{d}, \sigma^2, \phi') \).

The role of the extra parameter \( \lambda \) is to simply rescale the random effects. The null value of \( \lambda \) is \( \lambda_0 \) such that \( \Lambda(\lambda_0) = \mathbf{I}_b \). Note that when \( \Lambda = \mathbf{I}_b \) the expanded model is reduced to the general linear mixed model with \( u = f \) and \( G = D \).

Recall from Chapter 2 that the mixed model equations for the general linear mixed model in (2.1) are given by

\[
\begin{bmatrix}
X' R^{-1} X & X' R^{-1} Z \\
Z' R^{-1} X & Z' R^{-1} Z + G^{-1}
\end{bmatrix}
\begin{bmatrix}
\hat{\tau} \\
\hat{u}
\end{bmatrix}
= 
\begin{bmatrix}
X' R^{-1} y \\
Z' R^{-1} y
\end{bmatrix}
\]

and can be written as

\[
C \tilde{\beta} = W' R^{-1} y
\]

where \( C \) is the coefficient matrix

\[
C = 
\begin{bmatrix}
X' R^{-1} X & X' R^{-1} Z \\
Z' R^{-1} X & Z' R^{-1} Z + G^{-1}
\end{bmatrix}
\]

\( \tilde{\beta} \) is the vector of estimated effects

\[
\tilde{\beta} = \begin{bmatrix} \hat{\tau} \\ \hat{u} \end{bmatrix}
\]

and \( W = [X \ Z] \). The mixed model equations can be solved using

\[
\tilde{\beta} = C^{-1} W' R^{-1} y
\]

and the solutions are given by

\[
\hat{\tau} = (X' H^{-1} X)^{-1} X' H^{-1} y \\
\hat{u} = GZ' P y
\]

The variance parameters to be estimated are \( \theta \) and \( \kappa = (\gamma', \sigma^2, \phi')' \).

For the expanded model in (5.2), the mixed model equations are given by

\[
\begin{bmatrix}
X' R^{-1} X & X' R^{-1} Z \Lambda \\
\Lambda' Z' R^{-1} X & \Lambda' Z' R^{-1} Z \Lambda + D^{-1}
\end{bmatrix}
\begin{bmatrix}
\hat{\tau} \\
\hat{f}
\end{bmatrix}
= 
\begin{bmatrix}
X' R^{-1} y \\
\Lambda' Z' R^{-1} y
\end{bmatrix}
\]

and can be written as

\[
C_p \tilde{\beta}_p = W_p' R^{-1} y
\]

where \( C_p \) is the coefficient matrix of the mixed model equations for the expanded model

\[
C_p = 
\begin{bmatrix}
X' R^{-1} X & X' R^{-1} Z \Lambda \\
\Lambda' Z' R^{-1} X & \Lambda' Z' R^{-1} Z \Lambda + D^{-1}
\end{bmatrix}
\]
\( \hat{\beta}_p \) as the vector of estimated effects for the expanded model

\[
\hat{\beta}_p = \begin{bmatrix} \hat{\tau} \\ \hat{f} \end{bmatrix}
\]

and \( W_p = \begin{bmatrix} X & Z \Lambda \end{bmatrix} \).

The expanded mixed model equations can be solved using

\[
\hat{\beta}_p = C_p^{-1} W_p R^{-1} y
\]

The matrix \( C_p^{-1} \) can be obtained using Result A.12

\[
C_p^{-1} = \begin{bmatrix} C_{pXX} & C_{pXZ} \\ C_{pZX} & C_{pZZ} \end{bmatrix} = \begin{bmatrix} (X'H^{-1}X)^{-1} & -(X'H^{-1}X)^{-1}X'H^{-1}Z\Lambda D \\ -D\Lambda'Z'H^{-1}X(X'H^{-1}X)^{-1}D - D\Lambda'Z'PZ\Lambda D \end{bmatrix}
\] (5.3)

The solutions of the mixed model equations for the expanded model are given by

\[
\hat{\tau} = (X'H^{-1}X)^{-1}X'y \\
\hat{f} = D\Lambda'Z'Py
\] (5.4)

The variance parameters to be estimated are \( \theta \) and \( \mathcal{K} = (\mathbf{\kappa}^*, \mathbf{\lambda}')' \) where \( \mathbf{\kappa}^* = (d', \sigma^2, \phi')' \).

Define a matrix \( M \) given by

\[
M = \begin{bmatrix} I_t & 0 \\ 0 & \Lambda \end{bmatrix}
\]

and note that

\[
C_p = M'CM \quad \hat{\beta} = M\hat{\beta}_p \quad W_p = WM
\]

and hence

\[
C^{-1} = MC_p^{-1}M'
\]

In a similar fashion to the EM algorithm we view the observed data \( y \) as incomplete when using the PXEM algorithm. Recall from Section 4.5 that for the REML EM algorithm for mixed models that the random effects \( u \) are thought of as missing data and the complete data is made up of the observed data \( y \) and the unobserved random effects \( u \). The REML PXEM algorithm for mixed models rescales the random effects by \( \Lambda \) and we view the rescaled random effects \( f \) as missing data. Hence the complete data is comprised of the observed data \( y \) and the unobserved rescaled random effects \( f \). The distribution of the complete data is given by

\[
\begin{bmatrix} y \\ f \end{bmatrix} \sim N \left( \begin{bmatrix} X\tau \\ 0 \end{bmatrix}, \theta \begin{bmatrix} Z\Lambda D\Lambda'Z' + R & Z\Lambda D \\ D\Lambda'Z' & D \end{bmatrix} \right)
\]
The joint log-likelihood function of the complete data can be written as the sum of the log-likelihood of the missing data $f$ and the conditional log-likelihood of the observed data $y$ given the missing data $f$, namely

$$\ell_c(\theta, \mathbf{K}; y, f) = \ell_c(\theta, \mathbf{K}; f) + \ell_c(\theta, \mathbf{K}; y|f)$$

Note that $\ell_c(\theta, \mathbf{K}; y|f)$ is not strictly a log-likelihood since $f$ is not observed.

As we did for the REML EM algorithm we evaluate the complete data log-likelihood at $\theta = \theta^{(m)}$ as given in (2.54). The complete data log-likelihood evaluated at $\theta = \theta^{(m)}$ is given by

$$\ell_c(\mathbf{K}) = \ell_c(\theta, \mathbf{K}; y, f, \theta^{(m)}) = \ell_c(\theta, \mathbf{K}; f, \theta^{(m)}) + \ell_c(\theta, \mathbf{K}; y|f, \theta^{(m)})$$  \hspace{1cm} (5.5)

It is this form of the complete data log-likelihood that we use in the PXEM algorithm to obtain an update for $\mathbf{K}$ and it will be denoted by $\ell_c(\mathbf{K})$.

The log-likelihood of the missing data $f$ evaluated at $\theta = \theta^{(m)}$, excluding constant terms, is given by

$$\ell_c(\theta, \mathbf{K}; f, \theta^{(m)}) = -\frac{1}{2} \left( b \log \theta^{(m)} + \log |\mathbf{D}| + \frac{\mathbf{f}' \mathbf{D}^{-1} \mathbf{f}}{\theta^{(m)}} \right)$$ \hspace{1cm} (5.6)

This is a function of $\mathbf{d}$ only and will be denoted by $\ell_c(\mathbf{d})$.

Using Result A.11, the conditional distribution of the observed data $y$ given the missing data $f$ is given by

$$y|f \sim \mathcal{N}(\mathbf{X} \tau + \mathbf{Z} \Lambda \mathbf{f}, \theta \mathbf{R})$$

where $\mathbf{R} = \sigma^2 \Sigma$. Hence the log-likelihood of $y|f$ evaluated at $\theta = \theta^{(m)}$, excluding any constant terms, is given by

$$\ell_c(\sigma^2, \phi, \lambda) = \ell(\tau, \mathbf{K}; y|f, \theta = \theta^{(m)}) = -\frac{1}{2} \left( n \log \theta^{(m)} + n \log \sigma^2 + \log |\Sigma| + \frac{(y - \mathbf{X} \tau - \mathbf{Z} \Lambda \mathbf{f})' \Sigma^{-1} (y - \mathbf{X} \tau - \mathbf{Z} \Lambda \mathbf{f})}{\sigma^2 \theta^{(m)}} \right)$$

$$= -\frac{1}{2} \left( n \log \theta^{(m)} + n \log \sigma^2 + \log |\Sigma| + \frac{\mathbf{e}' \Sigma^{-1} \mathbf{e}}{\sigma^2 \theta^{(m)}} \right)$$  \hspace{1cm} (5.7)

This is a function of $\sigma^2$, $\phi$ and $\lambda$ only and will be denoted by $\ell_c(\sigma^2, \phi, \lambda)$.

Combining (5.6) and (5.7) allows the complete data log-likelihood in (5.5)

$$\ell_c(\mathbf{K}) = \ell_c(\mathbf{d}) + \ell_c(\sigma^2, \phi, \lambda) = -\frac{1}{2} \left( b \log \theta^{(m)} + \log |\mathbf{D}| + \frac{\mathbf{f}' \mathbf{D}^{-1} \mathbf{f}}{\theta^{(m)}} \right) - \frac{1}{2} \left( n \log \theta^{(m)} + n \log \sigma^2 + \log |\Sigma| + \frac{\mathbf{e}' \Sigma^{-1} \mathbf{e}}{\sigma^2 \theta^{(m)}} \right)$$  \hspace{1cm} (5.8)
The E step for the PXEM algorithm for REML estimation for mixed models computes the conditional expectation of the complete data log-likelihood in (5.8) given the incomplete data at $K^{(m)} = (\kappa^{(m)}', \chi_0')'$. We call this the PX-E step, and it is given by

$$E[\ell_c(K)|y_2; \theta^{(m)}, K^{(m)}] = E[\ell_c(d)|y_2; \theta^{(m)}, K^{(m)}] + E[\ell_c(\sigma^2, \phi, \lambda)|y_2; \theta^{(m)}, K^{(m)}]$$

For simplicity we will use the operand $E_{PX}[ ]^{(m)}$ to denote the conditional expectation of a function, given $y_2$ at the current fit for $\theta = \theta^{(m)}$ and $K = K^{(m)}$. Hence the PX-E step can be written as

$$E_{PX}[\ell_c(K)]^{(m)} = E_{PX}[\ell_c(d)]^{(m)} + E_{PX}[\ell_c(\sigma^2, \phi, \lambda)]^{(m)} \quad (5.9)$$

Note that the PX-E step consists of two parts; one is a function of $d$ and the other is a function of $\sigma^2, \phi$ and $\lambda$. We refer to these two parts as the PX-E step for $d$ and the PX-E step for $\sigma^2, \phi$ and $\lambda$.

First we will calculate the PX-E step for $d$, that is the conditional expectation of the log-likelihood of $f$ as given in (5.6), given $y_2$ at the current fit for $\theta = \theta^{(m)}$ and $K = K^{(m)}$.

$$E_{PX}[\ell_c(d)]^{(m)} = E[\ell_c(d)|y_2; \theta^{(m)}, K^{(m)}] = -\frac{1}{2} \left( b \log \theta^{(m)} + \log |D| + \frac{E_{PX}[f'D^{-1}f]^{(m)}}{\theta^{(m)}} \right) \quad (5.10)$$

To calculate $E_{PX}[f'D^{-1}f]^{(m)} = E[f'D^{-1}f|y_2; \theta^{(m)}, K^{(m)}]$ we need the distribution of $f$ conditional on $y_2$. The joint distribution of $f$ and $y_2$ is given by

$$\begin{bmatrix} f \\ y_2 \end{bmatrix} \sim N \left( \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \theta \begin{bmatrix} D & \Lambda'Z'L_2 \\ L'_2ZAD & L'_2HL_2 \end{bmatrix} \right)$$

Using Result A.11, the conditional distribution of $f$ given $y_2$ is

$$f|y_2 \sim N(D\Lambda'Z'Py, \theta(D - D\Lambda'Z'PZAD))$$

$$\sim N(\tilde{f}, \theta C_p ZZ) \quad (5.11)$$

using (5.4) and (5.3). Therefore using Result A.10

$$E[f'D^{-1}f|y_2] = \theta \text{tr}(C_p ZZ D^{-1}) + \tilde{f}'D^{-1}\tilde{f}$$

Evaluating this conditional expectation at $\theta^{(m)}$ and $K^{(m)} = (\kappa^{(m)}', \chi_0')'$ we get

$$E_{PX}[f'D^{-1}f]^{(m)} = E[f'D^{-1}f|y_2; \theta^{(m)}, K^{(m)}] = \theta^{(m)}\text{tr}(C^{ZZ(m)}D^{-1}) + \tilde{u}^{(m)}D^{-1}\tilde{u}^{(m)}$$

Note that this is equivalent to $E_{EM}[u'D^{-1}u]^{(m)}$.

We can now write the PX-E step for $d$ in (5.10) as

$$E_{PX}[\ell_c(d)]^{(m)} = -\frac{1}{2} \left( b \log \theta^{(m)} + \log |D| + \text{tr}(C^{ZZ(m)}D^{-1}) + \frac{\tilde{u}^{(m)}D^{-1}\tilde{u}^{(m)}}{\theta^{(m)}} \right) \quad (5.12)$$
Now we will calculate the PX-E step for $\sigma^2$, $\phi$ and $\lambda$, that is the conditional expectation of the log-likelihood of $y$ given $f$ conditional on $y_2$ at the current fit for $\theta = \theta^{(m)}$ and $\mathcal{K} = \mathcal{K}^{(m)}$.

\[
E_{ PX}[\ell_c(\sigma^2, \phi, \lambda)]^{(m)} = E[\ell_c(\sigma^2, \phi, \lambda)|y_2; \theta^{(m)}, \mathcal{K}^{(m)}] = -\frac{1}{2} \left( n \log \theta^{(m)} + n \log \sigma^2 + \log |\Sigma| + \frac{E_{ PX}[e'\Sigma^{-1}e]^{(m)}}{\sigma^2\theta^{(m)}} \right) \tag{5.13}
\]

To calculate $E_{ PX}[e'\Sigma^{-1}e]^{(m)} = E[e'\Sigma^{-1}e|y_2; \theta^{(m)}, \mathcal{K}^{(m)}]$ we need the distribution of $e$ conditional on $y_2$. From (4.17) we have

\[e|y_2 \sim N(\bar{e}, \theta W C^{-1}W')\]

Note that

\[WC^{-1}W' = WMC_p^{-1}M'W' = W_p C_p^{-1}W'\]

and hence

\[e|y_2 \sim N(\bar{e}, \theta W_p C_p^{-1}W_p')\]

Therefore, using Result A.10

\[E[e'\Sigma^{-1}e] = \theta \text{tr}(W_p C_p^{-1}W_p'\Sigma^{-1}) + \bar{e}'\Sigma^{-1}\bar{e}\]

Evaluating this conditional expectation at $\theta^{(m)}$ and $\mathcal{K}^{(m)} = (\kappa^* = \kappa^{(m)} ', \lambda^* = \lambda_0)'$ gives us

\[E_{ PX}[e'\Sigma^{-1}e]^{(m)} = E[e'\Sigma^{-1}e|y_2; \theta^{(m)}, \mathcal{K}^{(m)}] = \theta^{(m)}\text{tr}(WC^{-1(m)}W'\Sigma) + \bar{e}^{(m)}'\Sigma^{-1}\bar{e}^{(m)}\]

Note that this is equivalent to $E_{ EM}[e'\Sigma^{-1}e]^{(m)}$.

We can now write the PX-E step for $\sigma^2$, $\phi$ and $\lambda$ in (5.13) as

\[
E_{ PX}[\ell_c(\sigma^2, \phi, \lambda)]^{(m)} = -\frac{1}{2} \left( n \log \theta^{(m)} + n \log \sigma^2 + \log |\Sigma| + \frac{\text{tr}(WC^{-1(m)}W'\Sigma)}{\sigma^2} + \frac{\bar{e}^{(m)}'\Sigma^{-1}\bar{e}^{(m)}}{\sigma^2\theta} \right) \tag{5.14}
\]

Combining (5.12) and (5.14), the PX-E step in (5.9) as

\[
E_{ PX}[\ell_c(\mathcal{K})]^{(m)} = E_{ PX}[\ell_c(\mathcal{D})]^{(m)} + E_{ PX}[\ell_c(\sigma^2, \phi, \lambda)]^{(m)} = -\frac{1}{2} \left( b \log \theta^{(m)} + \log |D| + \text{tr}(C^{ZZ(m)}D^{-1}) + \frac{\bar{u}^{(m)}'D^{-1}\bar{u}^{(m)}}{\theta^{(m)}} \right) \tag{5.15}
\]
The M step for the PXEM algorithm for REML estimation for mixed models, referred to as the PX-E step, requires maximisation of the conditional expectation calculated in the PX-E step in (5.15) with respect to the variance parameters \( d, \sigma^2, \phi \) and \( \lambda \), to obtain updates of these variance parameters. The PX-E step in (5.9) is separated into 2 parts, the PX-E step for \( d \) as given in (5.12), and the PX-E step for \( \sigma^2, \phi \) and \( \lambda \) as given in (5.14). So when we differentiate with respect to the variance parameters in the PX-M step, we consider the 2 parts separately; the PX-E step for \( d \) and the PX-E step for \( \sigma^2, \phi \) and \( \lambda \) is differentiated with respect to \( d \) and the PX-E step for \( \sigma^2, \phi \) and \( \lambda \) is differentiated with respect to \( \sigma^2, \phi \) and \( \lambda \).

The PX-M step for \( \sigma^2 \) entails differentiating 5.12 with respect to \( d_{ij} \).

\[
\frac{\partial E_{PX}[\ell_c(K)]^{(m)}}{\partial d_{ij}} = E_{PX}[\frac{\partial \ell_c(d)}{\partial d_{ij}}] = \frac{1}{2} \left( \frac{\partial \log |D|}{\partial d_{ij}} + \frac{\partial \text{tr}(C^{ZZ(m)} D^{-1})}{\partial d_{ij}} + 1 \frac{\partial u^{(m)}_i D_i^{-1} \hat{d}_{ij} D_i^{-1} u^{(m)}_i}{\partial \hat{\sigma}^{(m)}} \right)
\]

This is equivalent to the M step for \( \gamma_{ij} \) in (4.20) with \( G \) and \( \gamma_{ij} \) replaced with \( D \) and \( d_{ij} \).

And so we can write the PX-M step for \( d_{ij} \) as

\[
\frac{\partial E_{PX}[\ell_c(K)]^{(m)}}{\partial d_{ij}} = \frac{1}{2} \left( \text{tr}(D_i^{-1} \hat{D}_{ij}) - \text{tr}(C^{ZZ(m)} D_i^{-1} \hat{D}_{ij} D_i^{-1}) - \frac{u^{(m)}_i D_i^{-1} \hat{d}_{ij} D_i^{-1} u^{(m)}_i}{\hat{\gamma}^{(m)}} \right)
\]

where \( \hat{D}_{ij} = \frac{\partial D}{\partial d_{ij}} \). To obtain an update, \( d_{ij}^{(m+1)} \), we equate (5.16) to zero and solve for \( d_{ij} \).

In the case where there is only one random term, the PX-M step for \( d_i \) is given by

\[
\frac{\partial E_{PX}[\ell_c(\Gamma)]^{(m)}}{\partial d_i} = \frac{1}{2} \left( \text{tr}(D^{-1} \hat{D}_i) - \text{tr}(C^{ZZ(m)} D^{-1} \hat{D}_i D^{-1}) - \frac{\hat{u}^{(m)} D^{-1} \hat{D}_i D^{-1} \hat{u}^{(m)}}{\hat{\theta}^{(m)}} \right)
\]

where \( \hat{D}_i = \frac{\partial D}{\partial d_i} \). To obtain an update, \( d_i^{(m+1)} \), we equate (5.17) to zero and solve for \( d_i \).

The PX-M step for \( \sigma^2 \) entails differentiating (5.14) with respect to \( \sigma^2 \).

\[
\frac{\partial E_{PX}[\ell_c(K)]^{(m)}}{\partial \sigma^2} = \frac{\partial E_{PX}[\ell_c(\sigma^2, \phi, \lambda)]}{\partial \sigma^2} = \frac{1}{2} \left( \frac{n}{\sigma^2} - \frac{\text{tr}(WC^{-1(m)}W'\Sigma^{-1})}{\sigma^4} - \frac{\hat{e}^{(m)} \Sigma^{-1} \hat{e}^{(m)}}{\sigma^4 \hat{\theta}} \right)
\]

The PXEM update for \( \sigma^2 \) is obtained by equating (5.18) to zero and solving for \( \sigma^2 \).

\[
\sigma^{2(m+1)} = \frac{1}{n} \left( \text{tr}(WC^{-1(m)}W'\Sigma^{-1}) + \frac{\hat{e}^{(m)} \Sigma^{-1} \hat{e}^{(m)}}{\hat{\theta}^{(m)}} \right)
\]

Note that the PXEM update for \( \sigma^2 \) is equivalent to the EM update for \( \sigma^2 \) as given in (4.24).
5.2. REML PXEM ALGORITHM FOR MIXED MODELS

The PX-M step for $\phi_i$ entails differentiating (5.14) with respect to $\phi_i$.

$$
\frac{\partial E_{\text{PX}}[\ell_c(K)]^{(m)}}{\partial \phi_i} = \frac{\partial E_{\text{PX}}[\ell_c(\sigma^2, \phi, \lambda)]}{\partial \phi_i}
= -\frac{1}{2} \left( \frac{\partial \log |\Sigma|}{\partial \phi_i} + \frac{1}{\sigma^2} \frac{\partial \text{tr}(WC^{-1(m)W'\Sigma^{-1})}}{\partial \phi_i} + \frac{1}{\sigma^2 \theta^{(m)}} \frac{\partial \tilde{e}^{(m)'\Sigma^{-1}e^{(m)}}}{\partial \phi_i} \right)
$$

This is equivalent to the M step for $\phi_i$ in (4.26). And so the PX-M step for $\phi_i$ is given by

$$
\frac{\partial E_{\text{PX}}[\ell_c(K)]^{(m)}}{\partial \phi_i} = -\frac{1}{2} \left( \text{tr}(\Sigma^{-1}\Sigma_i) - \text{tr}\left(\frac{WC^{-1(m)W'\Sigma^{-1}1(\Sigma^{-1}e^{(m)})}}{\sigma^2} \right) - \frac{\tilde{e}^{(m)'\Sigma^{-1}e^{(m)}}}{\sigma^2 \theta^{(m)}} \right)
$$

(5.20)

To obtain an update, $\phi_i^{(m+1)}$, we equate (5.20) to zero and solve for $\phi_i$.

The PX-M step for $\lambda_{ij}$ entails differentiating the PX-E step for $\sigma^2$, $\phi$ and $\lambda$ with respect to $\lambda_{ij}$. Rather than differentiating (5.14) we will differentiate the PX-E step as it is given in (5.13).

$$
\frac{\partial E_{\text{PX}}[\ell_c(K)]^{(m)}}{\partial \lambda_{ij}} = \frac{\partial E_{\text{PX}}[\ell_c(\sigma^2, \phi, \lambda)]^{(m)}}{\partial \lambda_{ij}}
= -\frac{1}{2} \left( \frac{\partial \text{tr}(\Sigma^{-1}\Sigma_i)}{\partial \lambda_{ij}} - \frac{1}{\sigma^2 \theta^{(m)}} \frac{\partial \text{tr}(\Sigma^{-1}1(\Sigma^{-1}e^{(m)})}{\partial \lambda_{ij}} \right)
= -\frac{1}{2} \left( \frac{1}{\sigma^2 \theta^{(m)}} E_{\text{PX}} \left[ \frac{\partial e^{(m)'\Sigma^{-1}e^{(m)}}}{\partial \lambda_{ij}} \right] \right)
$$

Note that

$$
e^{(m)'\Sigma^{-1}e} = (y - X\tau - Z\Lambda f)'\Sigma^{-1}(y - X\tau - Z\Lambda f)
= (y - X\tau)'\Sigma^{-1}(y - X\tau) - 2f'\Lambda'Z'\Sigma^{-1}(y - X\tau) + f'\Lambda'Z'\Sigma^{-1}Z\Lambda f
$$

and hence, using Result A.1

$$
\frac{\partial e^{(m)'\Sigma^{-1}e}}{\partial \lambda_{ij}} = -2f'\dot{\Lambda}'_{ij}Z'\Sigma^{-1}(y - X\tau) + 2f'\dot{\Lambda}'_{ij}Z'\Sigma^{-1}Z\Lambda f
$$

where $\dot{\Lambda}_{ij} = \frac{\partial \Lambda}{\partial \lambda_{ij}}$. Therefore the PX-M step for $\lambda_{ij}$ becomes

$$
\frac{\partial E_{\text{PX}}[\ell_c(K)]^{(m)}}{\partial \lambda_{ij}} = \frac{1}{\sigma^2 \theta^{(m)}} \left( E_{\text{PX}}[f'\dot{\Lambda}'_{ij}Z'\Sigma^{-1}(y - X\tau)]^{(m)} - E_{\text{PX}}[f'\dot{\Lambda}'_{ij}Z'\Sigma^{-1}Z\Lambda f]^{(m)} \right)
$$

(5.21)

The PXEM update for $\lambda_{ij}$ is obtained by equating (5.21) to zero and solving for $\lambda_{ij}$.

$$
E_{\text{PX}}[f'\dot{\Lambda}'_{ij}Z'\Sigma^{-1}Z\Lambda f]^{(m)} = E_{\text{PX}}[f'\dot{\Lambda}'_{ij}Z'\Sigma^{-1}(y - X\tau)]^{(m)}
$$

(5.22)

Note that

$$
\Lambda = \sum_{k=1}^{b} \sum_{l=1}^{b} \lambda_{kl}\hat{\Lambda}_{kl}
$$

(5.23)
and hence
\[ f' \hat{A}_{ij} Z' \Sigma^{-1} Z \Lambda f = \sum_{k=1}^{b} \sum_{l=1}^{b} \lambda_{kl} f' \hat{A}_{ij} Z' \Sigma^{-1} Z \Lambda_{kl} f \]

Therefore (5.22) can be written as
\[
\begin{align*}
&\sum_{k=1}^{b} \sum_{l=1}^{b} \lambda_{kl} E_{PX}[f' \hat{A}_{ij} Z' \Sigma^{-1} Z \Lambda_{kl} f]^{(m)} = E_{PX}[f' \hat{A}_{ij} Z' \Sigma^{-1} (y - X\tau)]^{(m)} \\
&\sum_{k=1}^{b} \sum_{l=1}^{b} \lambda_{kl} E_{PX}[\text{tr}(f' \hat{A}_{ij} Z' \Sigma^{-1} Z \Lambda_{kl} f f')]^{(m)} = E_{PX}[\text{tr}(f' \hat{A}_{ij} Z' \Sigma^{-1} (y - X\tau) f')]^{(m)} \\
&\sum_{k=1}^{b} \sum_{l=1}^{b} \lambda_{kl} \text{tr}(E_{PX}[\hat{A}_{ij} Z' \Sigma^{-1} Z \Lambda_{kl} f f'])^{(m)} = \text{tr}(E_{PX}[\hat{A}_{ij} Z' \Sigma^{-1} (y - X\tau) f'])^{(m)} \\
&\sum_{k=1}^{b} \sum_{l=1}^{b} \lambda_{kl} \text{tr}(\hat{A}_{ij} Z' \Sigma^{-1} Z \Lambda_{kl} E_{PX}[f f'])^{(m)} = \text{tr}(\hat{A}_{ij} Z' \Sigma^{-1} E_{PX}[(y - X\tau) f'])^{(m)}
\end{align*}
\]

(5.24)

To calculate \( E_{PX}[f f']^{(m)} = E[f f'| y_2; \theta^{(m)}, K^{(m)}] \) and \( E_{PX}[(y - X\tau) f']^{(m)} = E[(y - X\tau) f'| y_2; \theta^{(m)}, K^{(m)}] \) we need to find the joint distribution of \( y - X\tau \) and \( f \) conditional on \( y_2 \). For simplicity we define \( y_* = y - X\tau \). The joint distribution of \( y_* \), \( f \) and \( y_2 \) is given by
\[
\begin{bmatrix}
  y_* \\
  f \\
  y_2
\end{bmatrix}
\sim N
\begin{bmatrix}
  0 \\
  0 \\
  0
\end{bmatrix}, \theta
\begin{bmatrix}
  H & Z\Lambda & H L_2 \\
  D\Lambda' Z' & D & D\Lambda' Z' L_2 \\
  L_2 ' H & L_2 ' Z \Lambda & L_2 ' H L_2
\end{bmatrix}
\]

Using Result A.11 the expected value of joint distribution of \( y_* \) and \( f \) conditional on \( y_2 \) is
\[
E \begin{bmatrix}
  y_* \\
  f
\end{bmatrix} \mid y_2 = \begin{bmatrix}
  H P y \\
  D\Lambda' Z' P y
\end{bmatrix}
\[
= \begin{bmatrix}
  \hat{y}_* \\
  \hat{f}
\end{bmatrix}
\]

where \( \hat{y}_* = y - X\hat{\tau} \), and the variance of the joint distribution of \( y_* \) and \( f \) conditional on \( y_2 \) is
\[
\text{Var} \begin{bmatrix}
  y_* \\
  f
\end{bmatrix} \mid y_2 = \theta
\begin{bmatrix}
  H - H P H & Z\Lambda D - H P Z \Lambda D \\
  D\Lambda' Z' - D\Lambda' Z' P H & D - D\Lambda' Z' P \Lambda D
\end{bmatrix}
\]

\[
= \theta
\begin{bmatrix}
  X C_p ^{XX} X' & -X C_p ^{XZ} \\
  -C_p ^{ZX} X' & C_p ^{ZZ}
\end{bmatrix}
\]

Therefore the joint distribution of \( y_* \) and \( f \) conditional on \( y_2 \) is
\[
\begin{bmatrix}
  y_* \\
  f
\end{bmatrix} \mid y_2 \sim N
\begin{bmatrix}
  \hat{y}_* \\
  \hat{f}
\end{bmatrix}, \theta
\begin{bmatrix}
  X C_p ^{XX} X' & -X C_p ^{XZ} \\
  -C_p ^{ZX} X' & C_p ^{ZZ}
\end{bmatrix}
\]
5.2. REML PXEM ALGORITHM FOR MIXED MODELS

Using Result A.11

\[ E[ff'|y_2] = \tilde{f} \tilde{f}' + \theta C_p^{ZZ} \]

Evaluating this conditional expectation at \( \theta^{(m)} \) and \( \mathcal{K}^{(m)} \) gives

\[ E_{PX}[ff'|y_2]^{(m)} = E[ff'|y_2; \theta^{(m)}, \mathcal{K}^{(m)}] \]
\[ = \tilde{u}^{(m)} \tilde{u}'^{(m)} + \theta^{(m)} C^{ZZ(m)} \]

Note that this is equivalent to \( E_{EM}[uu']^{(m)} \). Again using Result A.11

\[ E[(y - X\tau)f'|y_2] = (y - \hat{X}\tau)\tilde{f}' - \theta X C_p^{XZ} \]

Evaluating this conditional expectation at \( \theta^{(m)} \) and \( \mathcal{K}^{(m)} \) gives

\[ E_{PX}[(y - X\tau)f'|y_2]^{(m)} = E[(y - X\tau)f'|y_2; \theta^{(m)}, \mathcal{K}^{(m)}] \]
\[ = (y - \hat{X}\tau^{(m)})\tilde{u}^{(m)} - \theta^{(m)} X C^{XZ(m)} \]

Note that this is equivalent to \( E_{EM}[(y - X\tau)u']^{(m)} \).

We can now write (5.24) as

\[ \sum_{k=1}^{b} \sum_{l=1}^{b} \lambda_{kl} \text{tr} \left( \hat{\Lambda}_{ij}^{'} Z' \Sigma^{-1} Z \hat{\Lambda}_{kl} (\tilde{u}^{(m)} \tilde{u}'^{(m)} + \theta^{(m)} C^{ZZ(m)}) \right) \]
\[ = \text{tr} \left( \hat{\Lambda}_{ij}^{'} Z' \Sigma^{-1} (y - \hat{X}\tau^{(m)}) \tilde{u}'^{(m)} - \theta^{(m)} X C^{XZ(m)} \right) \]

which can be rearranged to give

\[ \sum_{k=1}^{b} \sum_{l=1}^{b} \lambda_{kl} \left( \tilde{u}'^{(m)} \hat{\Lambda}_{ij}^{'} Z' \Sigma^{-1} Z \hat{\Lambda}_{kl} \tilde{u}^{(m)} + \theta^{(m)} \text{tr}(\hat{\Lambda}_{ij}^{'} Z' \Sigma^{-1} Z \hat{\Lambda}_{kl} C^{ZZ(m)}) \right) \]
\[ = \tilde{u}'^{(m)} \hat{\Lambda}_{ij}^{'} Z' \Sigma^{-1} (y - \hat{X}\tau^{(m)}) - \theta^{(m)} \text{tr}(\hat{\Lambda}_{ij}^{'} Z' \Sigma^{-1} X C^{XZ(m)}) \]

This can be written as

\[ \sum_{k=1}^{b} \sum_{l=1}^{b} \lambda_{kl} a_{ij,k,l}^{(m)} = b_{ij}^{(m)} \]

which is equivalent to

\[ a_{ij}'^{(m)} \lambda = b_{ij}^{(m)} \]

where \( a_{ij,k,l}^{(m)} = \{a_{ij,k,l}\}^{(m)} \)

\[ a_{ij,k,l}^{(m)} = E_{PX}[f_{ij,k,l}' Z' \Sigma^{-1} Z \hat{\Lambda}_{kl} f]'^{(m)} \]
\[ = E_{EM}[u_{ij,k,l}' Z' \Sigma^{-1} Z \hat{\Lambda}_{kl} u]'^{(m)} \]
\[ = \tilde{u}'^{(m)} \hat{\Lambda}_{ij}^{'} Z' \Sigma^{-1} Z \hat{\Lambda}_{kl} \tilde{u}^{(m)} + \theta^{(m)} \text{tr}(\hat{\Lambda}_{ij}^{'} Z' \Sigma^{-1} Z \hat{\Lambda}_{kl} C^{ZZ(m)}) \]
and
\[
\begin{align*}
b_{ij}^{(m)} &= E_{PX}[\dot{\Lambda}_{ij}Z'\Sigma^{-1}(y - X\tau)]^{(m)} \\
&= E_{EM}[u^\prime\dot{\Lambda}_{ij}Z'\Sigma^{-1}(y - X\tau)]^{(m)} \\
&= \tilde{u}^{(m)}\dot{\Lambda}_{ij}Z'\Sigma^{-1}(y - X\hat{\tau}^{(m)}) - \theta^{(m)}\text{tr}(\dot{\Lambda}_{ij}Z'\Sigma^{-1}XCXZ^{(m)})
\end{align*}
\]

If \(a_{ij}^{(m)}'\lambda = b_{ij}^{(m)}\) then
\[
A^{(m)}\lambda = b^{(m)} \tag{5.25}
\]

where \(b^{(m)} = \{b_{ij}^{(m)}\}\) and \(A^{(m)} = \{a_{ij,kl}^{(m)}\}\). The PXEM update for \(\lambda\) is obtained by equating (5.25) to zero and solving for \(\lambda\), that is
\[
\lambda^{(m+1)} = A^{-1(m)}b^{(m)} \tag{5.26}
\]

To obtain the update for \(G\) we form \(D^{(m+1)}\) and \(\Lambda^{(m+1)}\) and calculate
\[
G^{(m+1)} = \Lambda^{(m+1)}D^{(m+1)}\Lambda^{(m+1)'}
\]
Chapter 6

Rates of Convergence

6.1 Introduction

Even if it can be proved theoretically that an iterative sequence will converge in the limit to the required point, a method will be practicable only if convergence occurs with some rapidity. In this chapter we investigate the rate of convergence for the various iterative schemes outlined in the previous chapters.

6.2 Convergence Criterion

Before we discuss the rate of convergence, we need to outline the criterion by which we define convergence. Since the REML solution is not known explicitly, the decision as to whether an iteration has converged is based on monitoring the iteration sequence. Searle et al. (1992) and Longford (1993) discuss different types of convergence criteria. The iteration sequence can be terminated when the changes in consecutive parameter estimates are small, when the changes in the log-likelihood are small, when the score vector is close to the zero vector, or a combination of these. ASReml assesses convergence by looking at the change in the REML log-likelihood. The lmer function in R look at the relative change in the variance parameter estimates. We have chosen to use a relative change in parameters convergence criterion as recommended by Schaeffer and Kennedy (1985) and used by Foulley and van Dyk (2000) since we are interested in comparing the performance of our iterative schemes with the PXEM algorithm as used by Foulley and van Dyk (2000). Convergence is achieved on the $m$th iteration if

$$\sqrt{\frac{\sum_{i=1}^{k}(\kappa_i^{(m)} - \kappa_i^{(m-1)})^2}{\sum_{i=1}^{k}(\kappa_i^{(m)})^2}} < 10^{-8}$$

(6.1)

where $\kappa$ is the vector of variance parameters of length $k$. 
6.3 Order of Convergence

The rate of convergence can be measured by comparing the improvement at each step to the improvement at the previous step, that is to measure the closeness of $\kappa^{(m+1)}$ to $\hat{\kappa}$ relative to the closeness of $\kappa^{(m)}$ to $\hat{\kappa}$.

Following the methodology outlined in Kreyszig (1993), let $\kappa^{(m+1)} = M(\kappa^{(m)})$ define an iterative method where the iteration function $M$ updates the current estimate $\kappa^{(m)}$ to the new value $\kappa^{(m+1)}$. Assume the sequence $\{\kappa^{(m)}\}$ converges to a solution $\hat{\kappa}$. Then $\kappa^{(m)} = \hat{\kappa} + \epsilon^{(m)}$ where $\epsilon^{(m)}$ is the error at the $m$th iteration.

A Taylor series represents the behaviour of a function near a given point. Assuming that $M(\hat{\kappa})$ is differentiable any number of times, a Taylor series expansion of $\kappa^{(m+1)} = M(\kappa^{(m)})$ at $\kappa = \hat{\kappa}$ gives

$$\kappa^{(m+1)} = M(\kappa^{(m)}) = M(\hat{\kappa}) + \epsilon^{(m)} + \frac{1}{2} M''(\hat{\kappa}) \epsilon^{(m)} + \frac{1}{3!} M'''(\hat{\kappa}) \epsilon^{(m)} + \ldots$$

(6.2)

where $M'(\hat{\kappa})$, $M''(\hat{\kappa})$ and $M'''(\hat{\kappa})$ represent the respective first, second and third derivatives of $M(\kappa^{(m)})$ with respect to $\kappa$. Note that $\kappa^{(m)}$ converges to $\hat{\kappa}$, hence $\hat{\kappa} = M(\hat{\kappa})$. Subtracting this from both sides of (6.2) gives

$$\kappa^{(m+1)} - \hat{\kappa} = M'(\hat{\kappa}) \epsilon^{(m)} + \frac{1}{2} M''(\hat{\kappa}) \epsilon^{(m)} + \frac{1}{3!} M'''(\hat{\kappa}) \epsilon^{(m)} + \ldots$$

$$\epsilon^{(m+1)} = M'(\hat{\kappa}) \epsilon^{(m)} + \frac{1}{2} M''(\hat{\kappa}) \epsilon^{(m)} + \frac{1}{3!} M'''(\hat{\kappa}) \epsilon^{(m)} + \ldots$$

(6.3)

Hence, the error at the $(m+1)$th iteration can be written as a function of the error at the previous iteration.

Once the relationship between $\epsilon^{(m+1)}$ and $\epsilon^{(m)}$ is known, then the order of convergence is determined by the exponent of $\epsilon^{(m)}$ in the first non-zero term on the right hand side of (6.3). In the case of convergence, $\epsilon^{(m)}$ is small for large $m$ and hence the order of convergence is a measure of the speed of convergence. The higher the order, the greater the speed at which the error diminishes and hence the faster the convergence. An iterative method of order 2 converges rapidly once the estimate is close to the REML solution, while a method of order 1 might take a large number of iterations even once the estimate is close to the REML solution.

If an iterative scheme is of order 1,

$$\epsilon^{(m+1)} \approx M'(\hat{\kappa}) \epsilon^{(m)}$$

it has linear convergence. For first order processes, $M'(\hat{\kappa})$ is often referred to as the matrix rate of convergence, or simply, the rate of convergence.

If an iterative scheme is of order 2,

$$\epsilon^{(m+1)} \approx \frac{1}{2} M''(\hat{\kappa}) \epsilon^{(m)}$$
it has quadratic convergence. The error is squared at each step, hence near the solution, the number of significant digits approximately doubles with each iteration.

In the numerical methods literature (for example Gill et al. (1981)), this is often written as: suppose an iterative sequence \( \{ \kappa^{(m)} \} \) converges to a solution \( \hat{\kappa} \). If there exists a positive \( \alpha \) and \( c \) such that

\[
\lim_{m \to \infty} \left| \frac{\hat{\kappa} - \kappa^{(m+1)}}{\hat{\kappa} - \kappa^{(m)}} \right| = \lim_{m \to \infty} \left| \frac{\epsilon^{(m+1)}}{\epsilon^{(m)}} \right| = c
\]

then the sequence is said to converge to \( \hat{\kappa} \) with order of convergence \( \alpha \). The number \( c \) is known as the asymptotic error constant. If \( \alpha = 1 \) the sequence converges linearly, if \( \alpha = 2 \), the sequence converges quadratically.

6.3.1 Newton-Raphson Algorithm

Recall from Section (3.1), when \( \theta = 1 \), the Newton-Raphson algorithm updating equation for \( \kappa \) is

\[
\kappa^{(m+1)} = \kappa^{(m)} + I_O(\kappa^{(m)})^{-1} U(\kappa^{(m)})
\]

Note that throughout this section we assume that \( \theta = 1 \) for simplicity. The results for the case when \( \theta \neq 1 \) and \( \sigma^2 = 1 \) are not included but can easily be obtained.

The iteration function is given by

\[
M(\kappa) = \kappa + I_O(\kappa)^{-1} U(\kappa)
\]

To determine the order of convergence for the Newton-Raphson algorithm, we need to find the exponent of the first non-vanishing term in the Taylor series expansion of \( \kappa^{(m+1)} = M(\kappa^{(m)}) \) about \( \hat{\kappa} \) as shown in (6.2). To do this we need to consider the derivatives of the iteration function \( M(\kappa) \) evaluated at \( \hat{\kappa} \).

\[
\frac{\partial M(\kappa)}{\partial \kappa_i} = s_i + \frac{\partial I_O(\kappa)^{-1}}{\partial \kappa_i} U(\kappa) + I_O(\kappa)^{-1} \frac{\partial U(\kappa)}{\partial \kappa_i}
\]

\[
= s_i - I_O(\kappa)^{-1} \frac{\partial I_O(\kappa)}{\partial \kappa_i} I_O(\kappa)^{-1} U(\kappa) + I_O(\kappa)^{-1} \frac{\partial U(\kappa)}{\partial \kappa_i}
\]

where \( s_i \) is a vector of length \( k \) consisting of all zeros except for a one in the \( i \)th position.

Note that \( \frac{\partial U(\kappa)}{\partial \kappa_i} \) is equivalent to the negative of the \( i \)th column of the observed information matrix \( I_O(\kappa) \), hence \( \frac{\partial U(\kappa)}{\partial \kappa_i} = -I_O(\kappa) s_i \). Therefore

\[
\frac{\partial M(\kappa)}{\partial \kappa_i} = s_i - I_O(\kappa)^{-1} \frac{\partial I_O(\kappa)}{\partial \kappa_i} I_O(\kappa)^{-1} U(\kappa) - I_O(\kappa)^{-1} I_O(\kappa) s_i
\]

\[
= s_i - I_O(\kappa)^{-1} \frac{\partial I_O(\kappa)}{\partial \kappa_i} I_O(\kappa)^{-1} U(\kappa) - s_i
\]

\[
= -I_O(\kappa)^{-1} \frac{\partial I_O(\kappa)}{\partial \kappa_i} I_O(\kappa)^{-1} U(\kappa)
\]  

(6.4)
To determine the order of convergence, we need to evaluate this derivative at the REML solution \( \hat{\kappa} \).

\[
\frac{\partial M(\hat{\kappa})}{\partial \kappa_i} = -\mathcal{I}_O(\hat{\kappa})^{-1}\frac{\partial \mathcal{I}_O(\hat{\kappa})}{\partial \kappa_i}\mathcal{I}_O(\hat{\kappa})^{-1}U(\hat{\kappa}) = 0
\]

since \( U(\hat{\kappa}) = 0 \). Hence the Newton-Raphson algorithm is at least of second order.

### 6.3.2 Fisher Scoring Algorithm

Recall from Section (3.1), the Fisher Scoring algorithm updating equation for \( \kappa \) is

\[
\kappa^{(m+1)} = \kappa^{(m)} + \mathcal{I}_E(\kappa^{(m)})^{-1}U(\kappa^{(m)})
\]

Hence the iteration function is given by

\[
M(\kappa) = \kappa + \mathcal{I}_E(\kappa)^{-1}U(\kappa)
\]

The first derivative of the iteration function \( M(\kappa) \) is given by

\[
\frac{\partial M(\kappa)}{\partial \kappa_i} = s_i + \frac{\partial \mathcal{I}_E(\kappa)^{-1}U(\kappa)}{\partial \kappa_i} + \mathcal{I}_E(\kappa)^{-1}\frac{\partial U(\kappa)}{\partial \kappa_i}
\]

\[
= s_i - \mathcal{I}_E(\kappa)^{-1}\frac{\partial \mathcal{I}_E(\kappa)}{\partial \kappa_i}\mathcal{I}_E(\kappa)^{-1}U(\kappa) + \mathcal{I}_E(\kappa)^{-1}\frac{\partial U(\kappa)}{\partial \kappa_i}
\]

\[
= s_i - \mathcal{I}_E(\kappa)^{-1}\frac{\partial \mathcal{I}_E(\kappa)}{\partial \kappa_i}\mathcal{I}_E(\kappa)^{-1}U(\kappa) - \mathcal{I}_E(\kappa)^{-1}\mathcal{I}_O(\kappa)s_i
\]

(6.5)

since \( \frac{\partial U(\kappa)}{\partial \kappa_i} = -\mathcal{I}_O(\kappa)s_i \). Note however the Fisher Scoring algorithm uses the expected information matrix \( \mathcal{I}_E(\kappa) \). As a result, the first and last terms in (6.5) do not cancel as it did in (6.4) for the Newton-Raphson algorithm.

Evaluating the first derivative at \( \hat{\kappa} \) gives

\[
\frac{\partial M(\hat{\kappa})}{\partial \kappa_i} = s_i - \mathcal{I}_E(\hat{\kappa})^{-1}\mathcal{I}_O(\hat{\kappa})s_i
\]

(6.6)

since \( U(\hat{\kappa}) = 0 \). Therefore \( \frac{\partial M(\hat{\kappa})}{\partial \kappa_i} \neq 0 \). The Fisher Scoring Algorithm is at least of first order and convergence will be approximately quadratic if the observed and expected information matrices are similar.

The result in (6.6) can be written in matrix notation where

\[
M'(\hat{\kappa}) = \frac{\partial M(\kappa)}{\partial \kappa} \bigg|_{\kappa = \hat{\kappa}} = \mathcal{I}_k - \mathcal{I}_E(\hat{\kappa})^{-1}\mathcal{I}_O(\hat{\kappa}) = \mathcal{I}_E(\hat{\kappa})^{-1}(\mathcal{I}_E(\hat{\kappa}) - \mathcal{I}_O(\hat{\kappa}))
\]

(6.7)

This is the relative difference between \( \mathcal{I}_E(\hat{\kappa}) \) and \( \mathcal{I}_O(\hat{\kappa}) \) and is called the iteration derivative in Jorgensen (1993) and Smyth (1996). It is a measure of the accuracy with which the observed information is approximated by the expected information.
6.3. ORDER OF CONVERGENCE

### 6.3.3 Average Information Algorithm

The convergence behaviour of the Average Information algorithm follows that of the Fisher Scoring Algorithm. The Average Information algorithm updating equation for $\kappa$ is

$$\kappa^{(m+1)} = \kappa^{(m)} + \mathcal{I}_A(\kappa^{(m)})^{-1} U(\kappa^{(m)})$$  \hspace{1cm} (6.8)

The first derivative of the iteration function, evaluated at $\hat{\kappa}$ is given by

$$\frac{\partial M(\hat{\kappa})}{\partial \kappa_i} = s_i - \mathcal{I}_A(\hat{\kappa})^{-1} \mathcal{I}_O(\hat{\kappa}) s_i$$  \hspace{1cm} (6.9)

Therefore $\frac{\partial M(\hat{\kappa})}{\partial \kappa_i} \neq 0$. The Average Information Algorithm is at least of first order and convergence will be approximately quadratic if the average information matrix is similar to the observed information matrix.

In matrix notation, this becomes

$$M'(\hat{\kappa}) = \mathcal{I}_k - \mathcal{I}_A(\hat{\kappa})^{-1} \mathcal{I}_O(\hat{\kappa})$$

$$= \mathcal{I}_A(\hat{\kappa})^{-1} (\mathcal{I}_A(\hat{\kappa}) - \mathcal{I}_O(\hat{\kappa}))$$  \hspace{1cm} (6.10)

This is the relative difference between $\mathcal{I}_A(\hat{\kappa})$ and $\mathcal{I}_O(\hat{\kappa})$ and is a measure of the accuracy with which the observed information is approximated by the average information.

### 6.3.4 EM Algorithm

Recall from Section (4.5), unlike the Newton-Raphson, Fisher Scoring and AI algorithms, there is not a general explicit updating equation for $\kappa$ for the EM algorithm. As for the other iterative schemes, the EM algorithm defines a mapping $\kappa^{(m+1)} = M(\kappa^{(m)})$. As described in Section (6.3), a Taylor series expansion of $\kappa^{(m+1)} = M(\kappa^{(m)})$ at $\kappa = \hat{\kappa}$ gives

$$\kappa^{(m+1)} = M(\hat{\kappa}) + M'(\hat{\kappa}) e^{(m)} + \frac{1}{2} M''(\hat{\kappa}) e^{(m)^2} + \frac{1}{3!} M'''(\hat{\kappa}) e^{(m)^3} + \ldots$$

The order of convergence is given by the first non-zero term after $M(\hat{\kappa})$. Typically $M'(\hat{\kappa})$ is non-zero and hence the EM algorithm is a first order algorithm (McLachlan and Krishnan (1997) and Meng and Rubin (1994)). Hence

$$\kappa^{(m+1)} - \hat{\kappa} = M(\hat{\kappa})(\kappa^{(m)} - \hat{\kappa})$$

Examples will be presented in the following chapter demonstrating that $M'(\hat{\kappa}) \neq 0$.

Dempster et al. (1977) show that the rate matrix $M'(\hat{\kappa})$ for the EM algorithm can be written in terms of information matrices, just as it is for the NR and AI algorithms in (6.7) and (6.10).

Recall from Section 4.1, the density function of the complete data can be written as

$$g_c(y_c; \kappa) = g_c(y_o, y_m; \kappa) = g(y_o; \kappa) f(y_o, y_m | y_o; \kappa)$$
where the conditional density of the complete data given the observed data is denoted by \( f(y_o; y_m|y_o; \kappa) \) and is equivalent to \( f(y_m|y_o; \kappa) \).

Hence the log-likelihood function for the complete data can be expressed as
\[
\log g_c(y_c; \kappa) = \log g(y_o; \kappa) + \log f(y_m|y_o; \kappa)
\]
\[
\ell_c(\kappa; y_c) = \ell(\kappa; y_o) + \ell_m(\kappa; y_m|y_o)
\]

where \( \ell_m(\kappa; y_m|y_o) \) is the conditional log-likelihood of the complete data given the observed data.

The score vector for the complete data is given by
\[
\frac{\partial \ell_c(\kappa; y_c)}{\partial \kappa} = \frac{\partial \ell(\kappa; y_o)}{\partial \kappa} + \frac{\partial \ell_m(\kappa; y_m|y_o)}{\partial \kappa}
\]
\[
S_c(\kappa; y_c) = S(\kappa; y_o) + S_m(\kappa; y_m|y_o)
\]

The observed information matrix for the complete data is given by
\[
-I_o(\kappa; y_c) = I_O(\kappa; y_o) + I_{O_m}(\kappa; y_m|y_o)
\]

The conditional expected information matrix for the complete data is obtained by taking the expectation over the conditional distribution of the complete data given the observed data \( f(y_m|y_o; \kappa) \), and is given by
\[
E \left[ -\frac{\partial^2 \ell_c(\kappa; y_c)}{\partial \kappa \partial \kappa'} \bigg| y_o \right] = E \left[ -\frac{\partial^2 \ell(\kappa; y_o)}{\partial \kappa \partial \kappa'} \right| y_o + E \left[ -\frac{\partial^2 \ell_m(\kappa; y_m|y_o)}{\partial \kappa \partial \kappa'} \bigg| y_o \right]
\]

This is called the “missing information principle” by Orchard and Woodbury (1972). \( I_{Ec}(\kappa; y_c) \) measures the complete information that one would expect to have if there were no missing data, \( I_O(\kappa; y_o) \) measures the information carried by the observed data, and \( I_{Em}(\kappa; y_m|y_o) \) measures the loss of information due to the missing data. Hence the observed information for the complete data \( y_c = (y_o', y_m')' \) is equal to the information carried by the observed data \( y_o \) plus that carried by the missing data \( y_m \).

Note that in Chapter (3), \( I_E \) is used to denote the expected information matrix, whereas here \( I_E \) is used to represent the conditional expected information matrix.

Theorem 4 in Dempster et al. (1977) relates these information matrices to \( M'(\hat{\kappa}) \), by considering a Taylor series expansion of \( Q'(\kappa_1; \kappa_2) = \frac{\partial Q(\kappa_1; \kappa_2)}{\partial \kappa_1} \) about \( \kappa_1 = \hat{\kappa} \) and \( \kappa_2 = \hat{\kappa} \).
\[
Q'(\kappa_1; \kappa_2) \approx Q'(\hat{\kappa}; \hat{\kappa}) + \frac{\partial Q'(\kappa_1; \kappa_2)}{\partial \kappa_1} \bigg|_{\kappa_1 = \hat{\kappa}, \kappa_2 = \hat{\kappa}} (\kappa_1 - \hat{\kappa}) + \frac{\partial Q'(\kappa_1; \kappa_2)}{\partial \kappa_2} \bigg|_{\kappa_1 = \hat{\kappa}, \kappa_2 = \hat{\kappa}} (\kappa_2 - \hat{\kappa})
\]

Evaluating this at \( \kappa_1 = \kappa^{(m+1)} \) and \( \kappa_2 = \kappa^{(m)} \) gives
\[
Q'(\kappa^{(m+1)}; \kappa^{(m)}) \approx Q'(\hat{\kappa}; \hat{\kappa}) + \frac{\partial Q'(\kappa_1; \kappa_2)}{\partial \kappa_1} \bigg|_{\kappa_1 = \hat{\kappa}, \kappa_2 = \hat{\kappa}} (\kappa^{(m+1)} - \hat{\kappa}) + \frac{\partial Q'(\kappa_1; \kappa_2)}{\partial \kappa_2} \bigg|_{\kappa_1 = \hat{\kappa}, \kappa_2 = \hat{\kappa}} (\kappa^{(m)} - \hat{\kappa})
\]

(6.12)
Recall from the definition of the EM algorithm in Section 4.2, $\kappa^{(m+1)}$ is chosen such that $Q(\kappa; \kappa^{(m)})$ is maximised. Therefore $Q'(\kappa^{(m+1)}; \kappa^{(m)}) = 0$.

Recall from (4.7) that

$$S(\kappa; y_o) = \frac{\partial}{\partial \kappa} E[\ell_c(\kappa; y_c) \mid y_o]$$

$$= \frac{\partial}{\partial \kappa} Q(\kappa; \kappa)$$

$$= Q'(\kappa; \kappa)$$

hence $Q'(\hat{\kappa}; \hat{\kappa}) = S(\hat{\kappa}; y_o) = 0$.

Consider

$$\frac{\partial Q'(\kappa_1; \kappa_2)}{\partial \kappa_1'} = \frac{\partial^2}{\partial \kappa_1 \partial \kappa_1'} Q(\kappa_1; \kappa_2)$$

$$= \frac{\partial^2}{\partial \kappa_1 \partial \kappa_1'} E[\ell_c(\kappa_1; y_c) \mid y_o; \kappa_2]$$

$$= E \left[ \frac{\partial^2 \ell_c(\kappa_1; y_c)}{\partial \kappa_1 \partial \kappa_1'} \mid y_o; \kappa_2 \right]$$

Hence

$$\frac{\partial Q'(\kappa_1; \kappa_2)}{\partial \kappa_1'} \bigg|_{\kappa_1=\hat{\kappa}} \bigg|_{\kappa_2=\hat{\kappa}} = E \left[ \frac{\partial^2 \ell_c(\hat{\kappa}; y_c)}{\partial \hat{\kappa} \partial \hat{\kappa}'} \mid y_o; \hat{\kappa} \right]$$

$$= -I_{E_c}(\hat{\kappa}; y_c)$$

Recall from (5.1) that $Q(\kappa; \kappa^{(m)}) = \ell(\kappa; y_o) + H(\kappa; \kappa^{(m)})$ where $H(\kappa; \kappa^{(m)}) = E[\log f(y_c \mid y_o; \kappa) \mid y_o; \kappa^{(m)}]$. Hence

$$\frac{\partial Q'(\kappa_1; \kappa_2)}{\partial \kappa_2'} = \frac{\partial^2}{\partial \kappa_1 \partial \kappa_2'} Q(\kappa_1; \kappa_2)$$

$$= \frac{\partial}{\partial \kappa_1} \frac{\partial}{\partial \kappa_2'} \left( \ell(\kappa_1; y_o) + H(\kappa_1; \kappa_2) \right)$$

$$= \frac{\partial^2}{\partial \kappa_1 \partial \kappa_2'} H(\kappa_1; \kappa_2)$$

Noting that

$$H(\kappa_1; \kappa_2) = E[\log f(y_c \mid y_o; \kappa_1) \mid y_o; \kappa_2]$$

$$= \int \log f(y_c \mid y_o; \kappa_1) f(y_c \mid y_o; \kappa_2) dy_c$$
Consider

\[
\frac{\partial^2}{\partial \kappa_1 \partial \kappa_2} H(\kappa_1; \kappa_2') = \frac{\partial^2}{\partial \kappa_1 \partial \kappa_2} \int \log f(y_c \mid y_o; \kappa_1) f(y_c \mid y_o; \kappa_2) \, dy_c \\
= \int \frac{\partial}{\partial \kappa_1} \log f(y_c \mid y_o; \kappa_1) \frac{\partial}{\partial \kappa_2} f(y_c \mid y_o; \kappa_2) \, dy_c \\
= \int \frac{\partial}{\partial \kappa_1} \log f(y_c \mid y_o; \kappa_1) \frac{\partial}{\partial \kappa_2} \int \log f(y_c \mid y_o; \kappa_2) f(y_c \mid y_o; \kappa_2) \, dy_c \\
= E \left[ \frac{\partial}{\partial \kappa_1} \log f(y_c \mid y_o; \kappa_1) \frac{\partial}{\partial \kappa_2} \log f(y_c \mid y_o; \kappa_2) \bigg| y_o; \kappa_2 \right] \tag{6.13}
\]

And

\[
\frac{\partial^2}{\partial \kappa_1 \partial \kappa_1'} H(\kappa_1; \kappa_2') = \frac{\partial^2}{\partial \kappa_1 \partial \kappa_1'} \int \log f(y_c \mid y_o; \kappa_1) f(y_c \mid y_o; \kappa_2) \, dy_c \\
= \int \frac{\partial^2}{\partial \kappa_1 \partial \kappa_1'} \log f(y_c \mid y_o; \kappa_1) f(y_c \mid y_o; \kappa_2) \, dy_c \\
= \int \frac{\partial}{\partial \kappa_1} \frac{\partial}{\partial \kappa_1'} f(y_c \mid y_o; \kappa_1) f(y_c \mid y_o; \kappa_2) \, dy_c \\
= \int \frac{\partial^2}{\partial \kappa_1 \partial \kappa_1'} f(y_c \mid y_o; \kappa_1) f(y_c \mid y_o; \kappa_2) \, dy_c \\
- \int \left( \frac{\partial}{\partial \kappa_1} f(y_c \mid y_o; \kappa_1) \right)^2 f(y_c \mid y_o; \kappa_2) \, dy_c \\
= \int \frac{\partial^2}{\partial \kappa_1 \partial \kappa_1'} f(y_c \mid y_o; \kappa_1) f(y_c \mid y_o; \kappa_2) \, dy_c \\
- E \left[ \left( \frac{\partial}{\partial \kappa_1} f(y_c \mid y_o; \kappa_1) \right)^2 \bigg| y_o; \kappa_2 \right] \tag{6.14}
\]

When \( \kappa_1 = \kappa_2 \), the first term in (6.14) becomes

\[
\int \frac{\partial^2}{\partial \kappa_1 \partial \kappa_1'} f(y_c \mid y_o; \kappa_1) f(y_c \mid y_o; \kappa_1) \, dy_c = \int \frac{\partial^2}{\partial \kappa_1 \partial \kappa_1'} f(y_c \mid y_o; \kappa_1) \, dy_c \\
= \frac{\partial^2}{\partial \kappa_1 \partial \kappa_1'} \int f(y_c \mid y_o; \kappa_1) \, dy_c \\
= \frac{\partial^2}{\partial \kappa_1 \partial \kappa_1'} 1 \\
= 0
\]
6.3. ORDER OF CONVERGENCE

Hence when $\kappa_1 = \kappa_2$, (6.14) becomes
\[
\frac{\partial^2}{\partial \kappa_1 \partial \kappa_1} H(\kappa_1; \kappa_1) = -E \left[ \left( \frac{\partial}{\partial \kappa_1} \log f(y_c | y_o; \kappa_1) \right)^2 \right] y_o; \kappa_1
\]
and (6.13) becomes
\[
\frac{\partial^2}{\partial \kappa_1 \partial \kappa_1} H(\kappa_1; \kappa_1) = E \left[ \frac{\partial}{\partial \kappa_1} \log f(y_c | y_o; \kappa_1) \frac{\partial}{\partial \kappa_1} \log f(y_c | y_o; \kappa_1) \right] y_o; \kappa_1
\]
Thus when $\kappa_1 = \kappa_2$
\[
\frac{\partial^2}{\partial \kappa_1 \partial \kappa_2} H(\kappa_1; \kappa_2) = -\frac{\partial^2}{\partial \kappa_1 \partial \kappa_1} H(\kappa_1; \kappa_1)
\]
Hence
\[
\frac{\partial Q'(\kappa_1; \kappa_2)}{\partial \kappa_2} \bigg|_{\kappa_1 = \hat{\kappa}, \kappa_2 = \hat{\kappa}} = \frac{\partial^2}{\partial \kappa_1 \partial \kappa_2} H(\kappa_1; \kappa_2) \bigg|_{\kappa_1 = \hat{\kappa}, \kappa_2 = \hat{\kappa}} = -\frac{\partial^2}{\partial \kappa_1 \partial \kappa_1} H(\kappa_1; \kappa_2) \bigg|_{\kappa_1 = \hat{\kappa}, \kappa_2 = \hat{\kappa}} = -\frac{\partial^2}{\partial \kappa_1 \partial \kappa_1} E[\ell_m(\kappa_1; y_m | y_o) | y_o; \kappa_2] \bigg|_{\kappa_1 = \hat{\kappa}, \kappa_2 = \hat{\kappa}} = E \left[ -\frac{\partial^2}{\partial \kappa_1 \partial \kappa_1} \ell_m(\kappa_1; y_m | y_o) \right] y_o; \kappa_2 \bigg|_{\kappa_1 = \hat{\kappa}, \kappa_2 = \hat{\kappa}} = \mathcal{I}_{Em}(\hat{\kappa}; y_m | y_o)
\]
This allows (6.12) to be written as
\[
0 \approx 0 - \mathcal{I}_{Ec}(\hat{\kappa}; y_c)(\kappa^{(m+1)} - \hat{\kappa}) + \mathcal{I}_{Em}(\hat{\kappa}; y_m | y_o)(\kappa^{(m)} - \hat{\kappa})
\]
Therefore
\[
\mathcal{I}_{Ec}(\hat{\kappa}; y_c)(\kappa^{(m+1)} - \hat{\kappa}) \approx \mathcal{I}_{Em}(\hat{\kappa}; y_m | y_o)(\kappa^{(m)} - \hat{\kappa}) \quad (\kappa^{(m+1)} - \hat{\kappa}) \approx \mathcal{I}_{Ec}(\hat{\kappa}; y_c)(\kappa^{(m)} - \hat{\kappa})^{-1} \mathcal{I}_{Em}(\hat{\kappa}; y_m | y_o)(\kappa^{(m)} - \hat{\kappa})
\]
Recall that for a first order algorithm, $(\kappa^{(m+1)} - \hat{\kappa}) \approx M'(\hat{\kappa})(\kappa^{(m)} - \hat{\kappa})$, hence the rate matrix for the EM algorithm is given by
\[
M'(\hat{\kappa}) = [\mathcal{I}_{Ec}(\hat{\kappa}; y_c)]^{-1} \mathcal{I}_{Em}(\hat{\kappa}; y_m | y_o)
\]
[\mathcal{I}_{Ec}(\hat{\kappa}; y_c)]^{-1} \mathcal{I}_{Em}(\hat{\kappa}; y_m | y_o) is a measure of the proportion of information that is missing by not observing $y_m$ in addition to $y_o$. The greater the proportion of information in $y_m$, the smaller the proportion of information in $y_o$, and hence the slower the convergence of the EM algorithm. Dempster et al. (1977) comment that the amount of information loss may vary across the components of $\kappa$, suggesting that certain components of $\kappa$ may approach $\hat{\kappa}$ rapidly, while other components may require many iterations.
Note from (6.11) that $I_{E_m}(\kappa; y_m|y_o) = I_{E_c}(\kappa; y_c) - I_O(\kappa; y_o)$, hence
\[
M'(\hat{\kappa}) = \left[ I_{E_c}(\hat{\kappa}; y_c) \right]^{-1} \left( I_{E_c}(\hat{\kappa}; y_c) - I_O(\hat{\kappa}; y_o) \right) = I_k - \left[ I_{E_c}(\hat{\kappa}; y_c) \right]^{-1} I_O(\hat{\kappa}; y_o)
\]

The REML EM algorithm for the estimation of variance parameters for mixed models discussed in Section 4.5 involves the conditional expectation of the complete data log-likelihood $\ell_c(\kappa)$ conditional on the part of the observed data that contains the information about the variance parameters $y_2$. Hence the REML EM conditional expected information matrix for the complete data is given by
\[
I_{E_c}(\kappa) = E\left[ -\frac{\partial^2 \ell_c(\kappa)}{\partial \kappa \partial \kappa'} \right] y_2 \quad (6.15)
\]
where the complete data log-likelihood is given by
\[
\ell_c(\kappa) = -\frac{1}{2} \left( b \log \theta^{(m)} + \log |G| + \frac{u'G^{-1}u}{\theta} \right) - \frac{1}{2} \left( n \log \theta^{(m)} + n \log \sigma^2 + \log |\Sigma| + \frac{e'\Sigma^{-1}e}{\sigma^2 \theta^{(m)}} \right)
\]
and the REML EM observed information matrix for the observed data is given by
\[
I_O(\kappa) = -\frac{\partial^2 \ell_R(\kappa; y_2)}{\partial \kappa \partial \kappa'} \quad (6.16)
\]
which is the observed information matrix used in the Newton Raphson algorithm discussed in Chapter 3.

Therefore the rate matrix for the REML EM algorithm can be written as
\[
M'(\hat{\kappa}) = I_k - I_{E_c}(\hat{\kappa})^{-1} I_O(\hat{\kappa})
\]
This is of a similar form to the rate matrices for the Fisher Scoring and Average Information rate matrices given in (6.7) and (6.10).

6.3.5 PXEM Algorithm

Recall from Chapter 5 that the PXEM algorithm expands the parameter space to $\mathcal{K} = (\kappa^*, \lambda)$, and that $\mathcal{K}$ is reduced to the original parameter $\kappa$ via the reduction function $R$, that is $\kappa = R(\mathcal{K})$. To investigate the rate of convergence of the PXEM algorithm, Liu et al. (1998) use the parameterisation $\mathcal{K} = (\kappa, \lambda) = (R(\kappa^*, \lambda), \lambda)$ so that $\kappa$ is the parameter being estimated by the output of each PXEM iteration.

The PXEM algorithm defines a mapping $\mathcal{K}^{(m+1)} = M(\mathcal{K}^{(m)})$ and like the EM algorithm, is a first order algorithm, hence
\[
\mathcal{K}^{(m+1)} - \hat{\mathcal{K}} = M'(\hat{\mathcal{K}})(\mathcal{K}^{(m)} - \hat{\mathcal{K}})
\]
where $\mathcal{K}^{(m)} = (\kappa^{(m)}, \lambda_0)$. Following the methodology used to determine the convergence of the EM algorithm in the previous section, it can shown that the rate matrix for the REML PXEM algorithm can be written as
\[
M'(\hat{\mathcal{K}}) = I_k - I_{E_c}(\hat{\mathcal{K}})^{-1} I_O(\hat{\mathcal{K}})
\]
where $K = k + b^2$ is the number of variance parameters in $K$, $I_{Ec}(\hat{K})$ is the REML PXEM conditional expected information matrix for the complete data evaluated at $K = (\hat{\kappa}, \lambda_0)$, and $I_{O}(\hat{K})$ is the REML PXEM observed information matrix for the observed data evaluated at $K = (\hat{\kappa}, \lambda_0)$.

The REML PXEM conditional expected information matrix for the complete data is given by

$$I_{Ec}(K) = \begin{bmatrix} I_{Ec}(\kappa, \kappa') & I_{Ec}(\kappa, \lambda') \\ I_{Ec}(\lambda, \kappa') & I_{Ec}(\lambda, \lambda') \end{bmatrix}$$

where

$$I_{Ec}(\kappa, \kappa') = E \left[ -\frac{\partial^2 \ell_c(K)}{\partial \kappa \partial \kappa'} \right] y_2$$

$$I_{Ec}(\kappa, \lambda') = I_{Ec}(\lambda, \kappa')' = E \left[ -\frac{\partial^2 \ell_c(K)}{\partial \kappa \partial \lambda'} \right] y_2$$

$$I_{Ec}(\lambda, \lambda') = E \left[ -\frac{\partial^2 \ell_c(K)}{\partial \lambda \partial \lambda'} \right] y_2$$

where $\ell_c(K)$ is the PXEM complete data log-likelihood given in (5.8),

$$\ell_c(K) = -\frac{1}{2} \left( b \log \theta^{(m)} + \log |D| + \frac{f'D^{-1}f}{\theta^{(m)}} \right) - \frac{1}{2} \left( n \log \theta^{(m)} + n \log \sigma^2 + \log |\Sigma| + \frac{e'\Sigma^{-1}e}{\sigma^2\theta^{(m)}} \right)$$

Note that $I_{Ec}(\kappa, \kappa')$ is equivalent to the REML EM conditional expected information matrix for the complete data in (9.11).

The REML PXEM observed information matrix for the observed data is given by

$$I_{O}(K) = \begin{bmatrix} I_{O}(\kappa, \kappa') & 0 \\ 0 & 0 \end{bmatrix}$$

where $I_{O}(\kappa, \kappa')$ is equivalent to the observed information matrix in (6.16).

Following the notation of Liu et al. (1998), we write

$$I_{Ec}(\hat{K})^{-1} = \begin{bmatrix} V_{\hat{\kappa}\hat{\kappa}} & V_{\hat{\kappa}\lambda_0} \\ V_{\lambda_0\hat{\kappa}} & V_{\lambda_0\lambda_0} \end{bmatrix}$$

and hence the rate matrix for the PXEM algorithm becomes

$$M'(\hat{K}) = I_K - I_{Ec}(\hat{K})^{-1} I_{O}(\hat{K})$$

$$= I_K - \begin{bmatrix} V_{\hat{\kappa}\hat{\kappa}} & V_{\hat{\kappa}\lambda_0} \\ V_{\lambda_0\hat{\kappa}} & V_{\lambda_0\lambda_0} \end{bmatrix} \begin{bmatrix} I_{O}(\hat{\kappa}, \hat{\kappa}') & 0 \\ 0 & 0 \end{bmatrix}$$

$$= I_K - \begin{bmatrix} V_{\hat{\kappa}\hat{\kappa}}I_{O}(\hat{\kappa}, \hat{\kappa}') & 0 \\ V_{\lambda_0\hat{\kappa}}I_{O}(\hat{\kappa}, \hat{\kappa}') & 0 \end{bmatrix}$$

$$= \begin{bmatrix} I_{k} - V_{\hat{\kappa}\hat{\kappa}}I_{O}(\hat{\kappa}, \hat{\kappa}') & 0 \\ -V_{\lambda_0\hat{\kappa}}I_{O}(\hat{\kappa}, \hat{\kappa}') & I_{b^2} \end{bmatrix}$$
Therefore

\[
\mathbf{K}^{(m+1)} - \hat{K} = M'(\hat{K})(\mathbf{K}^{(m)} - \hat{K})
\]

\[
\begin{bmatrix}
\mathbf{K}^{(m+1)} - \hat{K} \\
\lambda^{(m+1)} - \lambda_0
\end{bmatrix} =
\begin{bmatrix}
I_k - V_{\hat{\kappa}\hat{\kappa}}I_O(\hat{\kappa}, \hat{\kappa}') & 0 \\
-V_{X_0\hat{\kappa}}I_O(\hat{\kappa}, \hat{\kappa}') & I_{E2}
\end{bmatrix}
\begin{bmatrix}
\mathbf{K}^{(m)} - \hat{K} \\
\lambda^{(m)} - \lambda_0
\end{bmatrix}
\]

\[
\mathbf{K}^{(m+1)} - \hat{K} = (I_k - V_{\hat{\kappa}\hat{\kappa}}I_O(\hat{\kappa}, \hat{\kappa}'))(\mathbf{K}^{(m)} - \hat{K})
\]

\[
\lambda^{(m+1)} - \lambda_0 = -V_{X_0\hat{\kappa}}I_O(\hat{\kappa}, \hat{\kappa}')(\mathbf{K}^{(m)} - \hat{K})
\]

because \(\lambda^{(m)} = \lambda_0\). Thus the convergence of the PXEM algorithm is determined by the convergence of \(\mathbf{K}^{(m)}\).

The rate matrix for \(\kappa\) for the PXEM algorithm is

\[
M'_{PX}(\hat{\kappa}) = I_k - V_{\hat{\kappa}\hat{\kappa}}I_O(\hat{\kappa})
\]

compared to the rate matrix for \(\kappa\) for the EM algorithm

\[
M'_{EM}(\hat{\kappa}) = I_k - I_{E(c)}(\hat{\kappa})^{-1}I_O(\hat{\kappa})
\]

Liu et al. (1998) describe \(V_{\hat{\kappa}\hat{\kappa}}\) as the complete-data variance of \(\kappa\) in PXEM, and \(I_{E(c)}(\hat{\kappa})^{-1}\) as the complete-data variance of \(\kappa\) in EM. They show that \(V_{\hat{\kappa}\hat{\kappa}} \geq I_{E(c)}(\hat{\kappa})^{-1}\) and hence the PXEM algorithm converges faster than the EM algorithm.

### 6.3.6 Measuring the Linear Rate of Convergence

Meng and Rubin (1994), Meng (1994) and McLachlan and Krishnan (1997) state that a measure of convergence for the EM algorithm is the convergence rate, which they defined as

\[
r = \lim_{m \to \infty} \frac{||\mathbf{K}^{(m+1)} - \mathbf{K}||}{||\mathbf{K}^{(m)} - \mathbf{K}||}
\]

where \(|| \cdot ||\) is the Euclidean norm. They state that under certain regularity conditions,

\[
r = \text{the largest eigenvalue of } M'(\hat{\kappa})
\]

Note that a large value of \(r\) implies slow convergence.

A number of authors, including Dempster et al. (1977), Meng and Rubin (1994), McLachlan and Krishnan (1997), Fraley (1999), Laird et al. (1987) and Sundberg (1976), have investigated the slow convergence of the EM algorithm. They state that convergence of the EM algorithm will occur provided \(0 \leq r < 1\). A convergence rate near 1 implies slow convergence, and a convergence rate near 0 implies nearly quadratic convergence. Dempster et al. (1977) and McLachlan and Krishnan (1997) state that exceptions to the convergence of the EM algorithm to a local maximum of the log-likelihood occur if \(M'(\hat{\kappa})\) has eigenvalues exceeding unity. An eigenvalue equal to one in the neighbourhood of \(\hat{\kappa}\) implies a ridge in the log-likelihood through \(\hat{\kappa}\).
Liu et al. (1998) show that the largest eigenvalue of the rate matrix for the PXEM algorithm is at least as small (and usually smaller than) as the largest eigenvalue of the rate matrix for the EM algorithm, and hence the PXEM algorithm dominates the EM algorithm in rate of convergence.

Horng (1987) present examples of slow convergence of the EM algorithm, including a linear mixed model. They use results from Ortega and Rheinboldt (1970) to show that for any iterative method $\kappa^{(m+1)} = M(\kappa^{(m)})$ that converges to a solution $\hat{\kappa}$,

\[
M \begin{cases}
\text{sublinear} & \text{convergence at } \hat{\kappa} \\
\text{linear} & \text{if } \rho(M'(\hat{\kappa})) = 1 \\
\text{superlinear} & \text{is in between 0 and 1} \\
 & = 0
\end{cases}
\]

where $\rho(M'(\hat{\kappa}))$ is the spectral radius of $M'(\hat{\kappa})$. The spectral radius of a matrix is the largest eigenvalue in absolute value of the matrix (Van Loan and Coleman, 1988). Note that Meng and Rubin (1994), Meng (1994) and McLachlan and Krishnan (1997) define the convergence rate as the largest eigenvalue of the rate matrix, not the largest eigenvalue in absolute value. This is because for the EM and PXEM algorithms, the eigenvalues of the rate matrix lie between 0 and 1 (McLachlan and Krishnan (1997), Fraley (1999)) and so there is no need to specify that it is the largest eigenvalue in absolute value.

The Fisher Scoring and AI algorithm are approximations of the Newton Raphson algorithm which is a second order algorithm. The price paid for using approximations of the observed information matrix that are easier to compute, is that the algorithms are first order algorithms. For these Newton-type algorithms the rate of convergence depends on the accuracy with which the observed information is approximated.

Smyth (1987), Osborne (1992), Smyth (1996), Wang (2007) discuss the convergence rate of the Fisher Scoring algorithm. Smyth (1987) and Smyth (1996) state that for a general iterative process define by $\kappa^{(m+1)} = M(\kappa^{(m)})$, the convergence of this process depends on the rate matrix $M'(\hat{\kappa})$. Ostrowski’s theorem (Ostrowski (1960), Ortega and Rheinboldt (1970)) asserts that a stationary point $\hat{\kappa}$ is a point of attraction of the iterative process if the spectral radius of the rate matrix is less than one. The spectral radius, which is given by the largest eigenvalue in absolute value of the rate matrix, is the rate of convergence. These results were presented with respect to the Fisher Scoring algorithm but can be applied to any first-order scheme. Hence we will redefine the convergence rate $r$ to be

\[
r = \text{the largest eigenvalue in absolute value of } M'(\hat{\kappa})
\]

We expect the AI algorithm to have a convergence rate close to zero and hence converge superlinearly, as it is an approximation to a second order scheme. However, the convergence rate is a measure of the rate of convergence within the neighbourhood of the REML solution, and the AI algorithm may diverge if the starting values are too far from the REML solution. The EM and PXEM algorithms are likely to have a convergence rate closer to one, with $r$ being smaller for PXEM than that for EM. Note that convergence of the EM and PXEM algorithms is guaranteed provided $0 \leq r < 1$. The theoretical rates of convergence for the AI, EM and PXEM algorithms will be examined for a range of models in Chapters 8 and 9.
In practice, $r$ is typically calculated by

$$
\lim_{m \to \infty} \frac{\|\kappa^{(m+1)} - \kappa^{(m)}\|}{\|\kappa^{(m)} - \kappa^{(m-1)}\|} = \lim_{m \to \infty} \sqrt{\frac{\sum_{i=1}^{k} (\kappa_i^{(m+1)} - \kappa_i^{(m)})^2}{\sum_{i=1}^{k} (\kappa_i^{(m)} - \kappa_i^{(m-1)})^2}}
$$

(6.17) (Meng, 1994, McLachlan and Krishnan, 1997, Schafer, 1997). We will call this the empirical rate of convergence and denote it by $r_e$.

In practice, it may not be feasible to calculate the theoretical convergence rate as it requires the calculation of the observed information matrix as well as the average information matrix or the conditional expected information matrix. The empirical convergence rate is easy to calculate and can be monitored as the iterations progress. In the following chapters we will calculate and compare the theoretical and empirical rates of convergence for the AI, EM and PXEM algorithms.

Meng (1994) noted that a large value of $r$ implies slow convergence. To be consistent with the common notion that the higher the value of the measure of convergence, the faster the algorithm convergence, they defined $s = 1 - r$ as the global speed of convergence. Thus $s$ is the smallest eigenvalue of

$$
S = I_k - M'(\hat{\kappa})
$$

where $S$ is a called the speed of convergence matrix.
Chapter 7

Improved Iterative Schemes

7.1 Introduction

In this chapter we introduce some improved iterative schemes. First we propose the use of hybrid schemes that use the EM or PXEM algorithm initially and then switch to the AI algorithm once the variance parameter estimates are in the neighbourhood of the true values. We then present ‘local’ EM and PXEM schemes that can be used in a hybrid scheme with the AI algorithm to update the random effects variance parameters only. We also look at non-iterative ways of obtaining variance parameter estimates that are in the neighbourhood of the true values to use as starting values.

7.2 Hybrid Schemes

The AI algorithm is a computationally convenient and efficient algorithm to use. However, when the starting values are too far from the REML estimates, the AI algorithm can step in the right direction but overshoot the solution or step in the wrong direction, producing variance parameter estimates that are outside of the parameter space and failing to converge to the REML solution. The EM and PXEM algorithms are good alternatives in the situations where the AI algorithm fails as they are robust to poor starting values and variance parameter estimates stay within the parameter space during iterations. However they can be very slow to converge. The variance parameter estimates rapidly approach the REML solution in the earlier iterations but then take a large number of iterations to actually converge to the REML solution. Therefore it is sensible to use a hybrid approach, using the EM or PXEM algorithm initially to get variance parameter estimates that are in the neighbourhood of the true values, and then switch to the AI algorithm using these estimates as improved starting values.

One approach is to use the AI algorithm unless it fails. We could define the AI algorithm as failing if an iteration produces variance parameter estimates that are outside the parameter space, where the parameter space is defined as all values of the variance
parameters that result in positive definite $G$ and $R$. However, in some cases the AI algorithm can produce variance parameter estimates outside this space in early iterations and still converge to a solution within the parameter space. In these situations, $G$ and/or $R$ are non-positive definite but $H = ZGZ' + R$ is usually still positive definite.

To ensure that a Newton type algorithm of the form $\kappa^{(m+1)} = \kappa^{(m)} + \mathcal{I}(\kappa^{(m)})^{-1}U(\kappa^{(m)})$ is guaranteed to converge to a maximum, $\mathcal{I}(\kappa^{(m)})$ needs to be positive definite (Harville, 1977, Gill et al., 1981, Dennis and Schnabel, 1996, Nocedal and Wright, 1999). And so for the AI algorithm, we require $\mathcal{I}_A(\kappa^{(m)})$ to be positive definite at each iteration. In the neighbourhood of the solution, the AI matrix will be positive definite. At points outside this neighbourhood, the AI matrix may be non-positive definite and the AI algorithm may diverge, converging to a stationary point that is not a local or global maximum and or may not converge at all. Hence we have chosen to terminate the AI algorithm if the AI matrix becomes non-positive definite. Note that by doing this we are defining the parameter space by the properties of the algorithm rather than the model, that is we define the parameter space as all values of the variance parameters that result in a positive definite AI matrix. This will allow the AI algorithm to converge in the cases mentioned above where the variance parameter estimates result in non-positive definite $G$ and/or $R$ in early iterations but then converge to a solution where $G$ and/or $R$ are positive definite.

If we use a hybrid scheme that uses the AI algorithm unless it fails, the first iteration starts with an AI update; that is we calculate the score and the AI matrix and then check to see if the AI matrix is positive definite. If the AI matrix is positive definite, the AI update is calculated. If the AI matrix is not positive definite, we say that the AI iteration has failed and we calculate an EM update. If AI fails for the first time after the first iteration, the iteration process is restarted with an EM iteration using the original starting values rather than the estimates from the previous AI iteration. This is done as the AI iteration before AI failed is likely to give estimates that are poor starting values as the AI algorithm may already be heading in the wrong direction. If the AI algorithm fails again after the iteration process has been restarted using an EM type update, the iteration process is resumed back at the iteration where the last EM type update was calculated, to make use of the progress that was made with the earlier EM update. This approach will be referred to as the hybrid scheme with the AI criterion.

Another approach is to investigate how close the algorithm is to convergence. Each iteration starts with an AI update. If the AI update is not in space (i.e. the estimates of the variance parameters do no result in a positive definite AI matrix), then an EM type update is calculated. If the AI update is in space, we then assess how close the AI update is to the REML solution. We consider two different ways of assessing how close the iterative process is to convergence.

One way to see how close the algorithm is to convergence is to check that the difference between the current estimates and the previous estimates is not too large. We do this by using a weaker form of the convergence criterion, that is, if

$$
\Delta^{(m)} = \sqrt{\frac{\sum (k_i^{(m)} - k_i^{(m-1)})^2}{\sum k_i^{(m)}^2}}
$$

(7.1)
7.3 Local Schemes

The convergence problems encountered when using the AI algorithm are often related to the parameters in the random effects variance matrix $G$, rather than the error variance matrix $R$ (Cullis et al., 2004). This lead us to the development of 'local' EM and PXEM schemes that only update the random effects variance parameters $\gamma$. These local schemes are a computationally cheaper than the full EM and PXEM algorithms. As discussed in Cullis et al. (2004), one of the motivations for the local PXEM scheme was the difficulty in implementing the full PXEM algorithm in ASReml due to the calculation of the off-diagonal terms of the inverse of the coefficient matrix $C$ of the mixed model equations, namely $C^{XZ}$. The local PXEM scheme does not require these terms.

We start by absorbing the fixed effects from the mixed model equations. The mixed model
equations are given by

\[ X' R^{-1} X \hat{\tau} + X' R^{-1} Z \tilde{u} = X' R^{-1} y \] (7.3)

\[ Z' R^{-1} X \hat{\tau} + (Z' R^{-1} Z + G^{-1}) \tilde{u} = Z' R^{-1} y \] (7.4)

Rearrange (7.3) to obtain

\[ \hat{\tau} = (X' R^{-1} X)^{-1} (X R^{-1} y - X' R^{-1} Z \tilde{u}) \]

and then substitute this into (7.4) to get

\[ Z' R^{-1} X (X' R^{-1} X)^{-1} (X R^{-1} y - X' R^{-1} Z \tilde{u}) + (Z' R^{-1} Z + G^{-1}) \tilde{u} = Z' R^{-1} y \]

Rearranging gives

\[ (Z' R^{-1} Z + G^{-1}) \tilde{u} - Z' R^{-1} X (X' R^{-1} X)^{-1} X' R^{-1} Z \tilde{u} = Z' R^{-1} y - Z' R^{-1} X (X' R^{-1} X)^{-1} X' R^{-1} y \]

and then

\[ (Z' (R^{-1} - R^{-1} X (X' R^{-1} X)^{-1} X' R^{-1}) Z + G^{-1}) \tilde{u} = Z' (R^{-1} - R^{-1} X (X' R^{-1} X)^{-1} X' R^{-1}) y \]

which can be written as

\[ (Z' S + G^{-1}) \tilde{u} = Z' S y \] (7.5)

where \( S = R^{-1} - R^{-1} X (X' R^{-1} X)^{-1} X' R^{-1} \).

Note that (7.5) can also be obtained via absorbing \( C_{XX} \) from the coefficient matrix of the mixed model equations using the absorption process outlined in 3.3.

We can consider \( Z'S \) to be a transformation of the data \( y \) that sweeps out the fixed effects. The expectation of \( Z'S y \) is given by

\[ E[Z'S y] = Z'S E[y] = Z'S X \tau \]

\[ = 0 \]

as \( S X = R^{-1} X - R^{-1} X (X' R^{-1} X)^{-1} X' R^{-1} X = R^{-1} X - R^{-1} X = 0 \). The variance of \( Z'S y \) is given by

\[ \text{Var}[Z'S y] = Z'S \text{Var}[y] S Z \]

\[ = \theta Z'S H S Z \]

\[ = \theta Z'S (Z G Z' + R) S Z \]

\[ = \theta (Z'S Z G Z' S Z + Z'S R S Z) \]

\[ = \theta (Z'S Z G Z' S Z + Z'S Z) \text{ (using Result A.8)} \]

\[ = \theta (\Omega G \Omega + \Omega) \]

where \( \Omega = Z'S Z \). Therefore the distribution of \( Z'S y \) is given by

\[ Z'S y \sim N(0, \theta (\Omega G \Omega + \Omega)) \]
Assuming that $\Omega$ is invertible, consider a modified transformation of the data

$$\frac{\Omega^{-1}Z'Sy}{\sqrt{\theta}}$$

that has expectation

$$E\left[\frac{\Omega^{-1}Z'Sy}{\sqrt{\theta}}\right] = \frac{\Omega^{-1}Z'SE[y]}{\sqrt{\theta}} = \frac{\Omega^{-1}Z'SX\tau}{\sqrt{\theta}} = 0$$

and variance

$$Var\left[\frac{\Omega^{-1}Z'Sy}{\sqrt{\theta}}\right] = \frac{\Omega^{-1}Z'SVar[y]SZ\Omega^{-1}}{\sqrt{\theta}} = \frac{\Omega^{-1}Z'SHSZ\Omega^{-1}}{\sqrt{\theta}} = \frac{\Omega^{-1}(\Omega G\Omega + \Omega)\Omega^{-1}}{\sqrt{\theta}} = G + \Omega^{-1}$$

and hence a distribution given by $\Omega^{-1}Z'Sy/\sqrt{\theta} \sim N(0, G + \Omega^{-1})$. This new variable has a simple variance structure that lends itself nicely to a linear model. However the matrix $\Omega = Z'SZ$ is not of full rank, and hence not invertible. Hence we approximate $\Omega$ by $\Omega^*$ which is of full rank and whose elements can easily be obtained from the external loop.

Result A.9 gives $C_{ZZ} = (G^{-1} + \Omega)^{-1}$ and hence

$$\Omega = (C_{ZZ})^{-1} - G^{-1}$$

If we approximate $C_{ZZ}$ by $C_{ZZ}^*$ where certain off-diagonal elements of $C_{ZZ}$ are set to zero, then $C_{ZZ}^* = (G^{-1} + \Omega^*)^{-1}$ and an appropriate full rank approximation to $\Omega$ is

$$\Omega^* = (C_{ZZ}^*)^{-1} - G^{-1}$$

Hence we approximate $\Omega$ by $\Omega^*$ which is of full rank and whose elements can easily be obtained from the external loop. Note that the form of $C_{ZZ}^*$ and hence the form of $\Omega^*$ depends on the model, particularly the structure of $G$.

These local schemes will be used within an iteration of the AI algorithm, which we will refer to as the current external loop. From the current external loop we will have the current values of the variance parameters, which we will denote as $\theta^{(m)}$ and $\kappa^{(m)} = (\gamma^{(m)}', a^{2(m)}, \phi^{(m)}')$. And then by absorbing the fixed effects we implicitly fix the estimates of the fixed effects. We define the local data as

$$v = \frac{\Omega^{(m)-1}Z'S^{(m)}y}{\sqrt{\theta^{(m)}}}$$

which has expectation

$$E[v] = \frac{\Omega^{(m)-1}Z'S^{(m)}E[y]}{\sqrt{\theta^{(m)}}} = \frac{\Omega^{(m)-1}Z'S^{(m)}X\tau}{\sqrt{\theta^{(m)}}} = 0$$
and variance

\[
\text{Var}[v] = \frac{\Omega^{(m)} Z' S^{(m)}\sqrt{\theta^{(m)}}}{\sqrt{\theta^{(m)}}} \text{Var}[y] = \frac{\Omega^{(m)} Z\Omega^{(m)} - 1}{\sqrt{\theta^{(m)}}} \\
= \frac{\Omega^{(m)} Z' S^{(m)}\theta H S^{(m)} Z\Omega^{(m)} - 1}{\sqrt{\theta^{(m)}}} \\
\approx \Omega^{(m)} Z' S^{(m)} HS^{(m)} Z\Omega^{(m)} - 1 (\text{assuming that } \theta \approx \theta^{(m)}) \\
= \Omega^{(m)} Z' S^{(m)} (ZGZ' + R) S^{(m)} Z\Omega^{(m)} - 1 \\
= \Omega^{(m)} Z' S^{(m)} ZGZ'S^{(m)} Z\Omega^{(m)} - 1 + \Omega^{(m)} - 1 Z'S^{(m)} RS^{(m)} Z\Omega^{(m)} - 1 \\
\approx \Omega^{(m)} Z' S^{(m)} ZGZ'S^{(m)} Z\Omega^{(m)} - 1 + \Omega^{(m)} - 1 Z'S^{(m)} Z\Omega^{(m)} - 1 \\
(\text{assuming that } R \approx R^{(m)} \text{ and using Result A.8}) \\
= \Omega^{(m)} - 1 G\Omega^{(m)} \Omega^{(m)} - 1 + \Omega^{(m)} - 1 \Omega^{(m)} \Omega^{(m)} - 1 \\
\approx G + \Omega^{(m)} - 1 (\text{assuming } \Omega^{(m)} \approx \Omega^{(m)})
\]

Hence the distribution of the local data is given by

\[v \sim N(0, G + \Omega^{(m)} - 1)\]

The local linear mixed model for the local data is given by

\[v = u_v + e_v\]

where the joint distribution of \(u_v\) and \(e_v\) is given by

\[
\begin{bmatrix}
u_v \\
e_v
\end{bmatrix} \sim N\left(\begin{bmatrix}0 \\
\end{bmatrix}, \begin{bmatrix}G & 0 \\
0 & \Omega^{(m)} - 1
\end{bmatrix}\right)
\]

The matrix \(\Omega^{(m)} - 1\) is fixed as it is evaluated at the values of the variance parameters from the external loop. The matrix \(G\) is a function of \(\gamma\).

The mixed model equation for this local model is given by

\[C_v u_v = \Omega^{(m)} v \quad \text{(7.6)}\]

where the coefficient matrix is given by \(C_v = \Omega^{(m)} + G^{-1}\).

The BLUP \(\tilde{u}_v\) is given by

\[\tilde{u}_v = C_v^{-1} \Omega^{(m)} v \quad \text{(7.7)}\]

Now that we have defined the local model, we will look at local EM and PXEM schemes to internally update \(\gamma\).
7.3. LOCAL SCHEMES

7.3.1 Local EM Algorithm

Recall that the EM algorithm treats the observed data as incomplete, and that in the case of linear mixed models, the random effects are thought of as the missing data. So for this local EM algorithm the complete data is made up of the observed local data $\mathbf{v}$ and the unobserved random effects $\mathbf{u}_v$. The distribution of the complete data is given by

$$
\begin{bmatrix}
\mathbf{v} \\
\mathbf{u}_v
\end{bmatrix} \sim \mathcal{N}
\left(
\begin{bmatrix}
0 \\
0
\end{bmatrix},
\begin{bmatrix}
\mathbf{G} + \Omega^{(m)^{-1}} & \mathbf{G} \\
\mathbf{G} & \mathbf{G}
\end{bmatrix}
\right)
$$

The complete data log-likelihood is given by

$$
\ell_c(\mathbf{v}, \mathbf{u}_v) = \ell_c(\mathbf{u}_v) + \ell_c(\mathbf{v}|\mathbf{u}_v)
$$

where the log-likelihood of $\mathbf{u}_v$ is given by

$$
\ell_c(\mathbf{u}_v) = -\frac{1}{2} \left( \log |\mathbf{G}| + \mathbf{u}_v' \mathbf{G}^{-1} \mathbf{u}_v \right)
$$

Using Result A.11, the conditional distribution of $\mathbf{v}$ given $\mathbf{u}_v$ is given by

$$
\mathbf{v}|\mathbf{u}_v \sim \mathcal{N}(\mathbf{u}_v, \Omega^{(m)^{-1}})
$$

and hence the log-likelihood of $\mathbf{v}|\mathbf{u}_v$ is given by

$$
\ell_c(\mathbf{v}|\mathbf{u}_v) = -\frac{1}{2} \left( \log |\Omega^{(m)^{-1}}| + (\mathbf{v} - \mathbf{u}_v)' \Omega^{(m)} (\mathbf{v} - \mathbf{u}_v) \right)
$$

The local E step computes the conditional expectation of the complete data log-likelihood, given the local data $\mathbf{v}$, at the current local fit of the variance parameters $\gamma = \gamma^{(n)}$, that is

$$
\mathbb{E}[\ell_c(\mathbf{v}, \mathbf{u}_v)|\mathbf{v}; \gamma^{(n)}] = \mathbb{E}[\ell_c(\mathbf{u}_v)|\mathbf{v}; \gamma^{(n)}] + \mathbb{E}[\ell_c(\mathbf{v}|\mathbf{u}_v)|\mathbf{v}; \gamma^{(n)}]
$$

which we denote by

$$
\mathbb{E}_{EM}[\ell_c(\mathbf{v}, \mathbf{u}_v)]^{(n)} = \mathbb{E}_{EM}[\ell_c(\mathbf{u}_v)]^{(n)} + \mathbb{E}_{EM}[\ell_c(\mathbf{v}|\mathbf{u}_v)]^{(n)}
$$

We will only calculate $\mathbb{E}_{EM}[\ell_c(\mathbf{u}_v)]^{(n)}$ as $\ell_c(\mathbf{u}_v)$ is the only part of the E step that is a function of $\gamma$.

$$
\mathbb{E}_{EM}[\ell_c(\mathbf{u}_v)]^{(n)} = -\frac{1}{2} \left( \log |\mathbf{G}| + \mathbb{E}_{EM}[\mathbf{u}_v' \mathbf{G}^{-1} \mathbf{u}_v]^{(n)} \right)
$$

To calculate $\mathbb{E}_{EM}[\mathbf{u}_v' \mathbf{G}^{-1} \mathbf{u}_v]^{(n)} = \mathbb{E}[\mathbf{u}_v' \mathbf{G}^{-1} \mathbf{u}_v|\mathbf{v}; \gamma^{(n)}]$ we need the conditional distribution of $\mathbf{u}_v$ given $\mathbf{v}$, which using Result A.11 is given by

$$
\mathbf{u}_v|\mathbf{v} \sim \mathcal{N}
\left(
\begin{bmatrix}
\mathbf{G}(\mathbf{G} + \Omega^{(m)^{-1}})^{-1} \mathbf{v} \\
\mathbf{G}(\mathbf{G} + \Omega^{(m)^{-1}})^{-1} \mathbf{G}
\end{bmatrix}
\right)
$$

$$
\sim \mathcal{N}(\tilde{\mathbf{u}}_v, \mathbf{C}_v^{-1})
$$
as
\[
G \left( G + \Omega^{(m)} \right)^{-1} v = G \left( G + \Omega^{(m)} \right)^{-1} \Omega^{(m)} \Omega^{(m)} v = \left( \Omega^{(m)} + G^{-1} \right)^{-1} \Omega^{(m)} v = \tilde{u}_v
\]
and
\[
G - G \left( G + \Omega^{(m)} \right)^{-1} G = (G^{-1} + \Omega^{(m)})^{-1} \quad \text{(Using Result A.2)}
\]

Therefore, using Result A.10,
\[
E_{EM}[u'G^{-1}u]^{(n)} = \text{tr}(C_{v}^{-1}G^{-1}) + \tilde{u}_v^{(n)'G^{-1}\tilde{u}_v^{(n)}}
\]
and hence
\[
E_{EM}[\ell_c(u_v)]^{(n)} = -\frac{1}{2} \left( \log |G| + \text{tr}(C_{v}^{-1}G^{-1}) + \tilde{u}_v^{(n)'G^{-1}\tilde{u}_v^{(n)}} \right) \quad (7.8)
\]

The local M step requires maximisation of the conditional expectation calculated in the local E step with respect to the variance parameters. For this local linear model, the variance parameter is $\gamma$ and so we only need to differentiate the part of the E step that relates to $\gamma$. That is why we only calculated $E_{EM}[\ell_c(u_v)]^{(n)}$ in the E step.

For simplicity of presentation we only consider a single random factor. The local M step for $\gamma_i$ is obtained by differentiating (7.8) by $\gamma_i$, giving
\[
\frac{\partial E_{EM}[\ell_c(u_v)]^{(n)}}{\partial \gamma_i} = -\frac{1}{2} \left( \frac{\partial \log |G|}{\partial \gamma_i} + \text{tr} \left( C_{v}^{-1} \frac{\partial G^{-1}}{\partial \gamma_i} \right) + \tilde{u}_v^{(n)'G^{-1}\tilde{u}_v^{(n)}} \right)
\]
\[
= -\frac{1}{2} \left( \text{tr}(G^{-1}G_i) - \text{tr}(C_{v}^{-1}G^{-1}G_iG^{-1}) - \tilde{u}_v^{(n)'G^{-1}\tilde{G}_iG^{-1}\tilde{u}_v^{(n)}} \right)
\]
using Result A.1. To obtain an internal update for $\gamma_i$ we equate (7.9) to zero and solve for $\gamma_i$. This will be illustrated by way of example in the following chapter.

### 7.3.2 Local PXEM Algorithm

We expand the local model in (7.6) to
\[
v = \Lambda f_v + e_v
\]
such that $u_v = \Lambda f_v$ where $\Lambda = \Lambda(\lambda)$ is a $b \times b$ invertible matrix and $f_v$ is a $b \times 1$ vector of rescaled random effects. The joint distribution of $f_v$ and $e_v$ is given by
\[
\begin{bmatrix} f_v \\ e_v \end{bmatrix} \sim N \left( \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} D & 0 \\ 0 & \Omega^{(m)}^{-1} \end{bmatrix} \right)
\]
7.3. LOCAL SCHEMES

\( D = D(d) \) is a symmetric positive definite variance matrix. Note that \( G = \Lambda D \Lambda' \). The variance parameters of the expanded model are \( K = (d', \lambda')' \).

The null value of \( \lambda \) is \( \lambda_0 \) such that \( \Lambda(\lambda) = I \), and when \( \Lambda = I \) the expanded local linear model is reduced to the local linear mixed model with \( u_v = f_v \) and \( G = D \).

The mixed model equation for the expanded local linear model is given by

\[
(\Lambda' \Omega^{(m)} \Lambda + D^{-1}) \tilde{f}_v = \Lambda' \Omega^{(m)} v
\]

where \( C_{v_p} = \Lambda' \Omega^{(m)} \Lambda + D^{-1} \).

The BLUP \( \tilde{f}_v \) is given by

\[
\tilde{f}_v = (\Lambda' \Omega^{(m)} \Lambda + D^{-1})^{-1} \Lambda' \Omega^{(m)} v
\]

In a similar fashion to the local EM algorithm, we view the local data \( v \) as incomplete. The complete data is made up of the local data \( v \) and the rescaled random effects \( f_v \).

The distribution of the complete data is given by

\[
\begin{bmatrix} v \\ f_v \end{bmatrix} \sim N\left( \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \Lambda D \Lambda' + \Omega^{(m)^{-1}} & \Lambda D \\ \Lambda D' & D \end{bmatrix} \right)
\]

The complete data log-likelihood is given by

\[
\ell_c(v, f_v) = \ell_c(f_v) + \ell_c(v|f_v)
\]

where the log-likelihood of \( f_v \) is given by

\[
\ell_c(f_v) = -\frac{1}{2} (\log |D| + f_v'D^{-1}f_v)
\]

Using Result A.11, the conditional distribution of \( v \) given \( f_v \) is given by

\[
v|f_v \sim N\left( \Lambda f_v, \Omega^{(m)^{-1}} \right)
\]

and hence the log-likelihood of \( v|f_v \) is given by

\[
\ell_c(v|f_v) = -\frac{1}{2} \left( \log |\Omega^{(m)^{-1}}| + (v - \Lambda f_v)' \Omega^{(m)} (v - \Lambda f_v) \right)
\]

\[
= -\frac{1}{2} \left( \log |\Omega^{(m)^{-1}}| + e'_v \Omega^{(m)} e_v \right)
\]

The local PX-E step computes the conditional expectation of the complete data log-likelihood given the local data \( v \) at \( K^{(n)} = (d' = \gamma^{(n)'}, \lambda' = \lambda_0')' \).

\[
E[\ell_c(v, f_v)|v; K^{(n)}] = E[\ell_c(f_v)|v; K^{(n)}] + E[\ell_c(v|f_v)|v; K^{(n)}]
\]

which we denote by

\[
E_{PX}[\ell_c(v, f_v)]^{(n)} = E_{PX}[\ell_c(f_v)]^{(n)} + E_{PX}[\ell_c(v|f_v)]^{(n)}
\]
The local PX-E step for \( d \) is given by
\[
E_{PX}[\ell_c(f_v)]^{(n)} = -\frac{1}{2} \left( \log |D| + E_{PX}[f'_v D^{-1} f_v]^{(n)} \right)
\]
To calculate \( E_{PX}[f'_v D^{-1} f_v]^{(n)} = \mathbb{E}[f'_v D^{-1} f_v | v; \mathcal{K}^{(n)}] \) we need the conditional distribution of \( f_v \) given \( v \), which using Result A.11 is given by
\[
f_v | v \sim N\left(D \Lambda' \left( \Lambda \Lambda' + \Omega^{(m)} \right)^{-1} v, D - D \Lambda' \left( \Lambda \Lambda' + \Omega^{(m)} \right)^{-1} \Lambda D \right)
\]
\[
\sim N(\tilde{f}_v, C_{v^{-1}})
\]
as
\[
D \Lambda' \left( \Lambda \Lambda' + \Omega^{(m)} \right)^{-1} v
\]
\[
= (\Lambda' \Omega^{(m)} \Lambda + D^{-1} \Lambda' \Omega^{(m)}) v \quad \text{(Using Result A.2)}
\]
and
\[
D - D \Lambda' \left( \Lambda \Lambda' + \Omega^{(m)} \right)^{-1} \Lambda D
\]
\[
= (\Lambda' \Omega^{(m)} \Lambda + D^{-1})^{-1} \quad \text{(Using Result A.2)}
\]
\[
= C_{v^{-1}}^{-1}
\]
Therefore using Result A.10
\[
\mathbb{E}[f'_v D^{-1} f_v | v] = \text{tr}(C_{v^{-1}} D^{-1}) + f'_v D^{-1} f_v
\]
Evaluating this conditional expectation at \( \mathcal{K}^{(n)} = (d' = \gamma^{(n)'}, \lambda' = \lambda^{(n)'} \) we get
\[
E_{PX}[f'_v D^{-1} f_v]^{(n)} = \mathbb{E}[f'_v D^{-1} f_v | v; \mathcal{K}^{(n)}] = \text{tr}(C_{v^{-1}}^{-1} D^{-1}) + \tilde{u}_v^{(n)' D^{-1} \tilde{u}_v^{(n)}}
\]
Hence the local PX-E step for \( d \) is given by
\[
E_{PX}[\ell_c(f_v)]^{(n)} = -\frac{1}{2} \left( \log |D| + \text{tr}(C_{v^{-1}}^{-1} D^{-1}) + \tilde{u}_v^{(n)' D^{-1} \tilde{u}_v^{(n)}} \right) \quad (7.10)
\]
The local PX-E step for \( \lambda \) is given by
\[
E_{PX}[\ell_c(v|f_v)]^{(n)} = -\frac{1}{2} \left( \log \left| \Omega^{(m)} \right|^{-1} \right) + E_{PX}[e'_v \Omega^{(m)} e_v]^{(n)}
\]
To calculate \( E_{PX}[e'_v \Omega^{(m)} e_v]^{(n)} = \mathbb{E}[e'_v \Omega^{(m)} e_v | v; \mathcal{K}^{(n)}] \) we need the conditional distribution of \( e_v \) given \( v \). The joint distribution of \( e_v \) and \( v \) is given by
\[
\begin{bmatrix} e_v \\ v \end{bmatrix} \sim N\left( \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \Omega^{(m)}^{-1} & \Omega^{(m)}^{(s^{-1})} \\ \Omega^{(m)}^{(s^{-1})} & \Lambda \Lambda' + \Omega^{(m)}^{-1} \end{bmatrix} \right)
\]
7.3. LOCAL SCHEMES

Using Result A.11, the conditional distribution of $e_v$ given $v$ is given by

$$
e_v | v \sim N\left( \Omega^{(m)-1} \left( \Lambda D \Lambda + \Omega^{(m)-1} \right)^{-1} v, \Omega^{(m)-1} - \Omega^{(m)-1} \left( \Lambda D \Lambda + \Omega^{(m)-1} \right)^{-1} \Omega^{(m)-1} \right)$$

as

$$
\Omega^{(m)-1} \left( \Lambda D \Lambda + \Omega^{(m)-1} \right)^{-1} v = \Omega^{(m)-1} \left( \Omega^{(m)} - \Omega^{(m)} \Lambda (D^{-1} + \Lambda' \Omega^{(m)} \Lambda)^{-1} \Lambda' \Omega^{(m)} \right) v \quad (\text{Using Result A.2})
$$

and

$$
\Omega^{(m)-1} - \Omega^{(m)-1} \left( \Lambda D \Lambda + \Omega^{(m)-1} \right)^{-1} \Omega^{(m)-1} = \Omega^{(m)-1} - \Omega^{(m)-1} \left( \Omega^{(m)} - \Omega^{(m)} \Lambda (D^{-1} + \Lambda' \Omega^{(m)} \Lambda)^{-1} \Lambda' \Omega^{(m)} \right) \Omega^{(m)-1} = \Lambda(D^{-1} + \Lambda' \Omega^{(m)} \Lambda)^{-1} \Lambda' = \Lambda C_{v^p}^{-1} \Lambda'
$$

Using Result A.10

$$
E[e'_v \Omega^{(m)} e_v | v] = \text{tr}(\Lambda C_{v^p}^{-1} \Lambda' \Omega^{(m)}) + e'_v \Omega^{(m)} e_v
$$

Evaluating this conditional expectation at $\mathcal{K}^{(n)} = (d' = \gamma^{(n)'}$, $\Lambda' = \Lambda^{(n)}$) we get

$$
E_{PX}[e'_v \Omega^{(m)} e_v | n] = E[e'_v \Omega^{(m)} e_v | v; \mathcal{K}^{(n)}] = \text{tr}(C_{v^p}^{-1(n)} \Omega^{(m)}) + e'_v \Omega^{(m)} e_v
$$

This allows us to write the local PX-E step for $\lambda$ as

$$
E_{PX}[\ell_c(v | f_v)](n) = -\frac{1}{2} \left( \log \left| \Omega^{(m)-1} \right| + \text{tr}(C_{v^p}^{-1(n)} \Omega^{(m)}) + e'_v \Omega^{(m)} e_v \right) \quad (7.11)
$$

The local PX-M step requires maximisation of the PX-E step with respect to the variance parameters $d$ and $\lambda$.

The local PX-M step for $d_i$ is obtained by differentiating the local PX-E step for $d$ in (7.10) by $d_i$.

$$
\frac{\partial E_{PX}[\ell_c(f_v)](n)}{\partial d_i} = -\frac{1}{2} \left( \frac{\partial \log |D|}{\partial d_i} + \text{tr} \left( C_{v^p}^{-1(n)} \frac{\partial D^{-1}}{\partial d_i} \right) + \hat{u}_v^{(n)'} \frac{\partial D^{-1}}{\partial d_i} \hat{u}_v^{(n)} \right) = -\frac{1}{2} \left( \text{tr}(D^{-1} \hat{D}_i) - \text{tr}(C_{v^p}^{-1(n)} D^{-1} \hat{D}_i \hat{D}^{-1}) - \hat{u}_v^{(n)'} D^{-1} \hat{D}_i \hat{D}^{-1} \hat{u}_v^{(n)} \right) \quad (7.12)
$$
Note that this is equivalent to the local M step for $\gamma_i$ in (7.9) with $G$ and $\gamma_i$ replaced with $D$ and $d_i$. To obtain an internal update for $d_i$ we equate the local PX-M step for $d_i$ in (7.12) to zero and solve for $d_i$. This will be illustrated by way of example in the following chapter.

The local PX-M step for $\lambda_{ij}$ is obtained by differentiating the local PX-E step for $\lambda$ in (7.11) by $\lambda_{ij}$

\[
\frac{\partial E_{PX} [\ell_c(v|f_v)](n)}{\partial \lambda_{ij}} = -\frac{1}{2} \left( \frac{\partial E_{PX} [e'_v \Omega^{*(m)}e_v](n)}{\partial \lambda_{ij}} \right)
\]

\[
= -\frac{1}{2} \left( E_{PX} \left[ \frac{\partial e'_v \Omega^{*(m)}e_v}{\partial \lambda_{ij}} \right] \right)
\]

Note that

\[
e'_v \Omega^{*(m)}e_v = (v - \Lambda f_v)' \Omega^{*(m)}(v - \Lambda f_v)
\]

\[
v' \Omega^{*(m)}v - 2f'_v \Lambda' \Omega^{*(m)}v + f'_v \Lambda' \Omega^{*(m)} \Lambda f_v
\]

and hence, using Result A.1

\[
\frac{\partial e'_v \Omega^{*(m)}e_v}{\partial \lambda_{ij}} = -2f'_v \Lambda'_{ij} \Omega^{*(m)}v + 2f'_v \Lambda'_{ij} \Omega^{*(m)} \Lambda f_v
\]

Therefore the local PX-M step for $\lambda_{ij}$ becomes

\[
\frac{\partial E_{PX} [\ell_c(v|f_v)](n)}{\partial \lambda_{ij}} = E_{PX} [f'_v \Lambda'_{ij} \Omega^{*(m)}v]^{(n)} - E_{PX} [f'_v \Lambda'_{ij} \Omega^{*(m)} \Lambda f_v]^{(n)}
\]

The local PXEM update for $\lambda_{ij}$ is obtained by equating to zero and solving for $\lambda_{ij}$

\[
E_{PX} [f'_v \Lambda'_{ij} \Omega^{*(m)} \Lambda f_v]^{(n)} = E_{PX} [f'_v \Lambda'_{ij} \Omega^{*(m)}v]^{(n)}
\]

\[
\sum_{k=1}^{b} \sum_{l=1}^{b} \lambda_{kl} E_{PX} [f'_v \Lambda'_{ij} \Omega^{*(m)} \Lambda_{kl} f_v]^{(n)} = E_{PX} [f'_v \Lambda'_{ij} \Omega^{*(m)}v]^{(n)} \quad \text{(Using (5.23))}
\]

\[
\sum_{k=1}^{b} \sum_{l=1}^{b} \lambda_{kl} E_{PX} [(\Lambda'_{ij} \Omega^{*(m)} \Lambda_{kl} f_v f'_v)]^{(n)} = E_{PX} [(\Lambda'_{ij} \Omega^{*(m)}v f'_v)]^{(n)}
\]

\[
\sum_{k=1}^{b} \sum_{l=1}^{b} \lambda_{kl} E_{PX} [(\Lambda'_{ij} \Omega^{*(m)} \Lambda_{kl} f_v f'_v)]^{(n)} = \text{tr}(\Lambda'_{ij} \Omega^{*(m)} E_{PX} [v f'_v])^{(n)} \quad \text{(7.13)}
\]

To calculate $E_{PX} [f_v f'_v]^{(n)} = E[f_v f'_v; \mathcal{K}^{(n)}]$ and $E_{PX} [v f'_v]^{(n)} = E[v f'_v; \mathcal{K}^{(n)}]$ we use Result A.11 to obtain

\[
E[f_v f'_v] = \tilde{f}_v \tilde{f}'_v + C_{v_p}^{-1}
\]

and

\[
E[v f'_v] = v \tilde{f}'_v
\]
and then evaluate these conditional expectations at $\mathbf{K}^{(n)} = (\mathbf{d}' = \gamma^{(n)'}, \lambda' = \lambda_0')$ to get

$$E_{P_X}[\mathbf{f}_v \mathbf{f}_v']^{(n)} = E[\mathbf{f}_v \mathbf{f}_v' | \mathbf{v}; \mathbf{K}^{(n)}]$$

$$= \tilde{u}_v^{(n)} \tilde{u}_v^{(n)'} + C_v^{-1(n)}$$

and

$$E_{P_X}[\mathbf{v} \mathbf{f}_v']^{(n)} = E[\mathbf{v} \mathbf{f}_v' | \mathbf{v}; \mathbf{K}^{(n)}]$$

$$= \mathbf{v} \mathbf{u}_v^{(n)'}$$

Hence we can write (7.13) as

$$\sum_{k=1}^{b} \sum_{l=1}^{b} \lambda_{kl} \text{tr}(\hat{A}'_{ij} \Omega^{s(m)} \hat{A}_{kl} (\hat{u}_v^{(n)} \hat{u}_v^{(n)'}) + C_v^{-1(n)}) = \text{tr}(\hat{A}'_{ij} \Omega^{s(m)} \hat{A}_{kl} \mathbf{v})$$

$$\sum_{k=1}^{b} \sum_{l=1}^{b} \lambda_{kl} (\tilde{u}_v^{(n)'} \hat{A}'_{ij} \Omega^{s(m)} \hat{A}_{kl} \tilde{u}_v^{(n)}) + \text{tr}(\hat{A}'_{ij} \Omega^{s(m)} \hat{A}_{kl} C_v^{-1(n)}) = \tilde{u}_v^{(n)'} \hat{A}'_{ij} \Omega^{s(m)} \mathbf{v}$$

$$\sum_{k=1}^{b} \sum_{l=1}^{b} \lambda_{kl} a^{(n)}_{v_{ij},kl} = b^{(n)}_{v_{ij}}$$

$$A_{v}^{(n)} \lambda = b^{(n)}_{v_{ij}}$$

where $a^{(n)}_{v_{ij}} = \{a^{(n)}_{v_{ij},kl}\}$,

$$a^{(n)}_{v_{ij},kl} = E_{P_X}[\mathbf{f}_v' \hat{A}'_{ij} \Omega^{s(m)} \hat{A}_{kl} \mathbf{f}_v]^{(n)}$$

$$= \tilde{u}_v^{(n)'} \hat{A}'_{ij} \Omega^{s(m)} \hat{A}_{kl} \tilde{u}_v^{(n)} + \text{tr}(\hat{A}'_{ij} \Omega^{s(m)} \hat{A}_{kl} C_v^{-1(n)})$$

and

$$b^{(n)}_{v_{ij}} = E_{P_X}[\mathbf{f}_v' \hat{A}'_{ij} \Omega^{s(m)} \mathbf{v}]^{(n)}$$

$$= \tilde{u}_v^{(n)'} \hat{A}'_{ij} \Omega^{s(m)} \mathbf{v}$$

If $a^{(n)}_{v_{ij}} \lambda = b^{(n)}_{v_{ij}}$ then

$$A_{v}^{(n)} \lambda = b^{(n)}_{v}$$

where $b^{(n)}_{v} = \{b^{(n)}_{v_{ij}}\}$ and $A_{v}^{(n)} = \{a^{(n)}_{v_{ij},kl}\}$. The local PXEM update for $\lambda$ is obtained by equating to zero and solving for $\lambda$, that is

$$\lambda^{(n+1)} = A_{v}^{-1(n)} b^{(n)}_{v}$$

To obtain the update for $\gamma$ we form $\mathbf{D}^{(n+1)}$ from $d^{(n+1)}_{v}$, and $\Lambda^{(n+1)}$ from $\lambda^{(n+1)}$ and calculate

$$G^{(n+1)} = \Lambda^{(n+1)} \mathbf{D}^{(n+1)} \Lambda^{(n+1)'}$$
7.3.3 Implementation of Local Schemes

Recall that for the composite algorithms described in 7.2 we begin with the AI algorithm and will only use an EM type algorithm if there are convergence problems with the AI algorithm. This extends to the local schemes. If the AI algorithm fails or if the AI update is not close enough to convergence, we use the local EM or PXEM scheme to update the random effects variance parameters and then switch back to the AI algorithm. We will refer to these schemes as the local EM/AI scheme and the local PXEM/AI scheme.

An iteration of the full EM or PXEM algorithm will always take the variance parameters closer to the true values. This is not true of the local schemes. The local schemes only update $\gamma$, with $\phi$ being held fixed at the value form the current external loop. Hence the local schemes are designed to be used only for a small number of iterations. The update for $\gamma$ from the first iteration of the local EM scheme is equivalent to the update for $\gamma$ from a full EM iteration, as detailed below. And so to benefit from this local EM scheme, at least two local iterations are needed. However the first iteration of the local PXEM scheme is not equivalent to an iteration of the full PXEM algorithm, since the prediction error covariance terms between fixed and random effects (ie. $C^{XZ}$) are ignored. And so a single local iteration may be all that is needed. The optimum number of internal local EM or PXEM iterations will be investigated in the following chapters.

To show that the update for $\gamma$ from the first iteration of the local EM scheme is equivalent to the update for $\gamma$ from a full EM iteration we consider the BLUP $\tilde{u}_v$ as given in (7.7),

$$
\tilde{u}_v = C_v^{-1}\Omega^*(m)v = C_v^{-1}\Omega^*(m)\Omega^*(m)^{-1}Z'S(m)y/\sqrt{\theta(m)} = C_v^{-1}Z'S(m)y/\sqrt{\theta(m)}
$$

Note that we approximate $\Omega = Z'SZ$ by $\Omega^* = (C^{ZZ^*})^{-1} - G^{-1}$ and hence we do not need to calculate $S$ to obtain $\Omega^*$. We can avoid the calculation of $S$ in $\tilde{u}_v$ by noting the following. Recall that the definition of the local data was motivated by absorbing the fixed effects from the mixed model equations and obtaining (7.5)

$$(Z'SZ + G^{-1})\hat{u} = Z'Sy$$

Result A.9 gives $C^{ZZ} = (\Omega + G^{-1})^{-1}$. Therefore

$$Z'Sy = (Z'SZ + G^{-1})\hat{u} = (\Omega + G^{-1})\hat{u} = (C^{ZZ^*})^{-1}\tilde{u} \approx (C^{ZZ^*})^{-1}\tilde{u}$$

Also note that $C_v = \Omega^*(m) + G^{-1} = (C^{ZZ^*}(m))^{-1}$ using Result A.9. Hence

$$\tilde{u}_v \approx C_v^{-1}((C^{ZZ^*}(m))^{-1}\tilde{u}/\sqrt{\theta(m)} = C^{ZZ^*}(m)(C^{ZZ^*}(m))^{-1}\tilde{u}/\sqrt{\theta(m)} = \tilde{u}/\sqrt{\theta(m)}$$
Recall that the local M step for $\gamma_i$ as given in (7.9) is
\[
\frac{\partial \mathbb{E}_{EM}[\ell_c(u,v)](n)}{\partial \gamma_i} = -\frac{1}{2} \left( \text{tr}(G^{-1}\dot{G}_i) - \text{tr}(C_v^{-1(n)}G^{-1}\dot{G}_iG^{-1}) - \tilde{u}_v^{(n)'}G^{-1}\dot{G}_iG^{-1}u_v^{(n)} \right)
\]
Noting that on the first local iteration $C_v^{(n)-1} = C^{ZZ^*(m)}$ and $\tilde{u}_v^{(n)} = \tilde{u}^{(m)}/\sqrt{\theta^{(m)}}$, this becomes
\[
\frac{\partial \mathbb{E}_{EM}[\ell_c(u,v)](1)}{\partial \gamma_i} = -\frac{1}{2} \left( \text{tr}(G^{-1}\dot{G}_i) - \text{tr}(C^{ZZ^*(m)}G^{-1}\dot{G}_iG^{-1}) - \frac{\tilde{u}^{(m)'}G^{-1}\dot{G}_iG^{-1}\tilde{u}^{(m)}}{\theta^{(m)}} \right)
\]
This is equivalent to the M step for $\gamma_i$ as given in (4.22) if $\text{tr}(C^{ZZ^*(m)}G^{-1}\dot{G}_iG^{-1}) = \text{tr}(C^{ZZ^*(m)}G^{-1}\dot{G}_iG^{-1})$. Recall that $C^{ZZ^*}$ is an approximation of $C^{ZZ}$ where certain off-diagonal elements are set to zero. The exact form of $C^{ZZ^*}$ depends on the model. In the following chapter it will be shown by example that the update for $\gamma$ from the first iteration of the local EM scheme is equivalent to the update for $\gamma$ from a full EM iteration.

### 7.4 Starting Values

The initial EM type iterations used in the hybrid schemes discussed in the previous section can be regarded as refining the starting values of the variance parameters before beginning the AI iterations. An alternative is to calculate good starting values for the variance parameters. Initial values may be obtained from a previous analysis of similar data, or they can be derived from the current data. Thisted (1988) suggest that a preliminary estimator such as a moment estimator can be used to get sufficiently close to the solution to ensure that the iterative scheme converges rapidly. Beal (1991) presented a computational strategy for obtaining initial estimates based on a general formulation of the method of moments for mixed models. However, we will develop a different method based on Henderson’s Method III, a method that was used to estimate variance components prior to development of REML.

Henderson’s Method III (HMIII), was introduced in Henderson (1953) and reformulated in matrix notation in Searle et al. (1992). This method estimates variance components using the sums of squares that result from fitting a linear model and various submodels of it. The sums of squares are computed as though the random effects are fixed effects and are then equated to their expected values which are derived under the mixed model. The estimates of the variance parameters are obtained by solving for these equations for the variance parameters. This will be illustrated by way of example in the following chapters.
Chapter 8

Variance Components Model

8.1 Introduction

In this chapter we investigate and compare the performance of the various iterative
schemes for a linear mixed model that contains a single set of uncorrelated random ef-
fects. The random effect variance structure is $G = \sigma^2_u I$ and the error variance structure
is $R = \sigma^2 I$ (note that $\theta = 1$).

We present the form of the updating equations for the variance parameters for the AI, EM
and PXEM algorithms and the local EM and local PXEM schemes. The performance of
the various iterative schemes is compared using the lamb weight data analysed by Harville
and Fenech (1985) and simulated incomplete block design data. Firstly, we compare the
performance of the AI, EM and PXEM algorithms, including the theoretical and empirical
convergence rates. For the data sets for which the AI algorithm does not converge, we
investigate the use of the hybrid schemes, including selection of the number of internal
local iterations for the local schemes and the cut-offs for the update and score criteria
for the hybrid schemes. We then use these results to develop an algorithm that is both
robust and converges rapidly.

8.2 Variance Components Model and Results

The variance components model is given by

$$y = X\tau + Zu + e$$  \hspace{1cm} (8.1)

where $y$ is a $n \times 1$ vector of observations, $\tau$ is a $t \times 1$ vector of fixed effects, $X$ is the
associated $n \times t$ design matrix of full column rank, $u$ is a $b \times 1$ vector containing a single
random effect, $Z$ is the associated $n \times b$ design matrix, and $e$ is a $n \times 1$ vector of residuals.
It is assumed that the joint distribution of $(u, e)$ is given by

$$\begin{bmatrix} u \\ e \end{bmatrix} \sim N \left( \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma^2_u I_b & 0 \\ 0 & \sigma^2 I_n \end{bmatrix} \right)$$

97
The variance parameters are the random effect variance component $\sigma_u^2$ and the error variance component $\sigma^2$, hence $\kappa = (\sigma_u^2, \sigma^2)$. Note that $\theta = 1$.

The REML log-likelihood for this model is given by

$$-\frac{1}{2} \left( b \log \sigma_u^2 + n \log \sigma^2 + \log |C| + y'Py \right)$$

The REML score for $\sigma_u^2$ is given by

$$U(\sigma_u^2) = -\frac{1}{2} \left( \frac{b}{\sigma_u^2} - \frac{\text{tr}(CZZ)}{\sigma_u^4} - \frac{\tilde{u}'\tilde{u}}{\sigma_u^4} \right)$$ \hfill (8.2)

The REML score for $\sigma^2$ is given by

$$U(\sigma^2) = -\frac{1}{2} \left( \frac{n}{\sigma^2} - \frac{\text{tr}(WC^{-1}W')}{\sigma^4} \right)$$

$$= -\frac{1}{2} \left( \frac{n - t - b}{\sigma^2} + \frac{\text{tr}(CZZ)}{\sigma_u^2 \sigma^2} - \tilde{e}'\tilde{e} \right)$$ \hfill (8.3)

since using (2.50) we have

$$\frac{\text{tr}(WC^{-1}W')}{\sigma^2} = t + b - \frac{\text{tr}(CZZ)}{\sigma_u^2}$$ \hfill (8.4)

The observed information matrix is given by

$$\mathcal{I}_O = \begin{bmatrix} \mathcal{I}_O(\sigma_u^2, \sigma_u^2) & \mathcal{I}_O(\sigma_u^2, \sigma^2) \\ \mathcal{I}_O(\sigma_u^2, \sigma^2) & \mathcal{I}_O(\sigma^2, \sigma^2) \end{bmatrix}$$

8.2.1 AI Algorithm

Given an initial estimate of $\kappa$, $\kappa^{(m)} = (\sigma_u^{2(m)}, \sigma^{2(m)})$, an update of $\sigma_u^2$ and $\sigma^2$ using the AI algorithm is given by

$$\begin{bmatrix} \sigma_u^{2(m+1)} \\ \sigma^{2(m+1)} \end{bmatrix} = \begin{bmatrix} \sigma_u^{2(m)} \\ \sigma^{2(m)} \end{bmatrix} + \mathcal{I}_A^{(m)-1} \begin{bmatrix} U(\sigma_u^{2(m)}) \\ U(\sigma^{2(m)}) \end{bmatrix}$$
8.2. VARIANCE COMPONENTS MODEL AND RESULTS

where $\mathcal{I}_A$ is partitioned as

$$\mathcal{I}_A = \begin{bmatrix} \mathcal{I}_A(\sigma_u^2, \sigma_u^2) & \mathcal{I}_A(\sigma_u^2, \sigma_y^2) \\ \mathcal{I}_A(\sigma_y^2, \sigma_u^2) & \mathcal{I}_A(\sigma_y^2, \sigma_y^2) \end{bmatrix}$$

where the elements are given by

$$\mathcal{I}_A(\sigma_u^2, \sigma_u^2) = \frac{1}{2} y' PZZ' PZZ' Py$$
$$\mathcal{I}_A(\sigma_u^2, \sigma_y^2) = \frac{1}{2} y' PZZ' P^2 y$$
$$\mathcal{I}_A(\sigma_y^2, \sigma_u^2) = \frac{1}{2} y' P^3 y$$

From Section 6.3.3, the convergence rate matrix of the AI algorithm is given by

$$M'_{AI}(\hat{\kappa}) = I - \mathcal{I}_A(\hat{\kappa})^{-1}\mathcal{I}_O(\hat{\kappa})$$

and the rate of convergence rate is given by the largest eigenvalue of $M'_{AI}(\hat{\kappa})$.

8.2.2 EM Algorithm

The M-step for $\sigma_u^2$ is given by

$$\frac{\partial E_{EM}[\ell_c(\kappa)]^{(m)}}{\partial \sigma_u^2} = -\frac{1}{2} \left( \frac{b}{\sigma_u^2} - \frac{\text{tr}(CZZ^{(m)})}{\sigma_u^4} - \frac{\hat{u}^{(m)}' \tilde{u}^{(m)}}{\sigma_u^4} \right)$$

Equating to zero and solving for $\sigma_u^2$, we obtain the EM update for $\sigma_u^2$,

$$\sigma_u^2^{(m+1)} = \frac{1}{b} \left( \text{tr}(CZZ^{(m)}) + \hat{u}^{(m)}' \tilde{u}^{(m)} \right) \quad (8.5)$$

The EM update for $\sigma^2$ is given by

$$\sigma^2^{(m+1)} = \frac{1}{n} \left( \text{tr}(WC^{-1(m)}W') + \tilde{\varepsilon}^{(m)}' \tilde{\varepsilon}^{(m)} \right)$$

$$= \frac{1}{n} \left( \sigma^2^{(m)}(t + b) - \frac{\sigma^2_u^{(m)} \text{tr}(CZZ^{(m)})}{\sigma_u^2^{(m)}} + \tilde{\varepsilon}^{(m)}' \tilde{\varepsilon}^{(m)} \right) \quad (8.6)$$

The rate of convergence of the EM algorithm is given by

$$M'_{EM}(\hat{\kappa}) = I - \mathcal{I}_{Ec}(\hat{\kappa})^{-1}\mathcal{I}_O(\hat{\kappa})$$

where

$$\mathcal{I}_{Ec}(\kappa) = E \left[ \frac{\partial^2 \ell_c(\kappa)}{\partial \kappa \partial \kappa'} \middle| y \right]$$
The REML EM complete data log-likelihood for the variance components model is given by

\[
\ell_c(\kappa) = -\frac{1}{2} \left( b \log \sigma_u^2 + \frac{u'u}{\sigma_u^2} \right) - \frac{1}{2} \left( n \log \sigma^2 + \frac{e'e}{\sigma^2} \right)
\]

The derivatives of \( \ell_c(\kappa) \) needed to calculate \( \mathbf{I}_{Ec}(\kappa) \) are

\[
\frac{\partial \ell_c(\kappa)}{\partial \sigma_u^2} = -\frac{1}{2} \left( \frac{b}{\sigma_u^2} - \frac{u'u}{\sigma_u^4} \right)
\]

\[
\frac{\partial^2 \ell_c(\kappa)}{\partial (\sigma_u^2)^2} = -\frac{1}{2} \left( -\frac{b}{\sigma_u^4} + \frac{2u'u}{\sigma_u^6} \right)
\]

\[
\frac{\partial^2 \ell_c(\kappa)}{\partial \sigma_u^2 \partial \sigma^2} = \frac{\partial^2 \ell_c(\kappa)}{\partial \sigma^2 \partial \sigma_u^2} = 0
\]

\[
\frac{\partial \ell_c(\kappa)}{\partial \sigma^2} = -\frac{1}{2} \left( \frac{n}{\sigma^4} - \frac{e'e}{\sigma^6} \right)
\]

\[
\frac{\partial^2 \ell_c(\kappa)}{\partial (\sigma^2)^2} = -\frac{1}{2} \left( -\frac{n}{\sigma^4} + \frac{2e'e}{\sigma^6} \right)
\]

Hence the elements of the REML EM conditional expected information matrix for the complete data are

\[
\mathbf{I}_{Ec}(\sigma_u^2, \sigma_u^2) = \mathbb{E} \left[ -\frac{\partial^2 \ell_c(\kappa)}{\partial (\sigma_u^2)^2} \right] y_2
\]

\[
= \frac{1}{2} \left( -\frac{b}{\sigma_u^4} + \frac{2 \mathbb{E}[u'u|y_2]}{\sigma_u^6} \right)
\]

\[
= \frac{1}{2} \left( -\frac{b}{\sigma_u^4} + \frac{2 \text{tr}(C^Z Z)}{\sigma_u^6} + \tilde{u}'\tilde{u} \right)
\]

\[
\mathbf{I}_{Ec}(\sigma_u^2, \sigma^2) = \mathbf{I}_{Ec}(\sigma^2, \sigma_u^2)
\]

\[
= \mathbb{E} \left[ -\frac{\partial^2 \ell_c(\kappa)}{\partial \sigma_u^2 \partial \sigma^2} \right] y_2
\]

\[
= 0
\]

\[
\mathbf{I}_{Ec}(\sigma^2, \sigma^2) = \mathbb{E} \left[ -\frac{\partial^2 \ell_c(\kappa)}{\partial (\sigma^2)^2} \right] y_2
\]

\[
= \frac{1}{2} \left( -\frac{n}{\sigma^4} + \frac{2 \mathbb{E}[e'e|y_2]}{\sigma^6} \right)
\]

\[
= \frac{1}{2} \left( -\frac{n}{\sigma^4} + \frac{2 \text{tr}(WC^{-1}W') + \tilde{e}'\tilde{e})}{\sigma^6} \right)
\]
8.2. VARIANCE COMPONENTS MODEL AND RESULTS

Evaluating the elements of $\mathbf{I}_{Ec}(\kappa)$ at $\kappa = \hat{\kappa}$ gives

$$
\mathbf{I}_{Ec}(\hat{\sigma}_u^2, \hat{\sigma}_e^2) = \frac{1}{2} \left( -\frac{b}{\hat{\sigma}_u^4} + \frac{2 (\text{tr}(\mathbf{C}^{ZZ}) + \hat{u}'\hat{u})}{\hat{\sigma}_u^6} \right)
$$

$$
= \frac{b}{2\hat{\sigma}_u^4} \quad (\text{since } b\hat{\sigma}_u^2 = \text{tr}(\mathbf{C}^{ZZ}) + \hat{u}'\hat{u})
$$

$$
\mathbf{I}_{Ec}(\hat{\sigma}_e^2, \hat{\sigma}_u^2) = \mathbf{I}_{Ec}(\hat{\sigma}_e^2, \hat{\sigma}_e^2) = 0
$$

$$
\mathbf{I}_{Ec}(\hat{\sigma}_u^2, \hat{\sigma}_u^2) = \frac{1}{2} \left( -\frac{n}{\hat{\sigma}_u^4} + \frac{2 (\text{tr}(\mathbf{W}^{-1}\mathbf{C}^{W'}) + \hat{e}'\hat{e})}{\hat{\sigma}_u^6} \right)
$$

$$
= \frac{n}{2\hat{\sigma}_u^4} \quad (\text{since } n\hat{\sigma}_u^2 = \text{tr}(\mathbf{W}^{-1}\mathbf{C}^{W'}) + \hat{e}'\hat{e})
$$

Therefore

$$
\mathbf{I}_{Ec}(\hat{\kappa}) = \begin{bmatrix}
\frac{b}{2\hat{\sigma}_u^4} & 0 \\
0 & \frac{n}{2\hat{\sigma}_u^4}
\end{bmatrix}
$$

and hence

$$
\mathbf{I}_{Ec}(\hat{\kappa})^{-1} = \begin{bmatrix}
\frac{2\hat{\sigma}_u^4}{b} & 0 \\
0 & \frac{2\hat{\sigma}_u^4}{n}
\end{bmatrix}
$$

Therefore the convergence rate matrix is given by

$$
\mathbf{M}'_{EM}(\hat{\kappa}) = \mathbf{I} - \mathbf{I}_{Ec}(\hat{\kappa})^{-1}\mathbf{I}_O(\hat{\kappa})
$$

$$
= \mathbf{I} - \begin{bmatrix}
\frac{2\hat{\sigma}_u^4}{b} & 0 \\
0 & \frac{2\hat{\sigma}_u^4}{n}
\end{bmatrix} \begin{bmatrix}
\mathbf{I}_O(\hat{\sigma}_u^2, \hat{\sigma}_e^2) & \mathbf{I}_O(\hat{\sigma}_e^2, \hat{\sigma}_u^2) \\
\mathbf{I}_O(\hat{\sigma}_e^2, \hat{\sigma}_e^2) & \mathbf{I}_O(\hat{\sigma}_u^2, \hat{\sigma}_e^2)
\end{bmatrix}
$$

and the rate of convergence is given by the largest eigenvalue of $\mathbf{M}'_{EM}(\hat{\kappa})$.

8.2.3 PXEM Algorithm

We expand the variance components model given in (8.1) to

$$
y = \mathbf{X}\tau + \mathbf{Z}\lambda \mathbf{f} + \mathbf{e}
$$

such that $\mathbf{u} = \lambda \mathbf{f}$ where $\mathbf{f}$ is a vector of rescaled random effects. The joint distribution of $\mathbf{f}$ and $\mathbf{e}$ is given by

$$
\begin{bmatrix}
\mathbf{f} \\
\mathbf{e}
\end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix}
0 \\
0
\end{bmatrix}, \begin{bmatrix}
\sigma_d^2 \mathbf{I}_b & 0 \\
0 & \sigma^2 \mathbf{I}_n
\end{bmatrix}\right)
$$

Note that

$$
\text{Var}(\mathbf{u}) = \lambda^2 \text{Var}(\mathbf{f})
$$

$$
\sigma_u^2 \mathbf{I}_b = \lambda^2 \sigma_d^2 \mathbf{I}_b
$$

$$
\sigma_u^2 = \lambda^2 \sigma_d^2
$$
The variance parameters of the expanded model are $\mathbf{K}' = (\mathbf{K}^\prime, \lambda)$ where $\mathbf{K}^\prime = (\sigma_u^2, \sigma^2)$. The null value of $\lambda$ is $\lambda_0 = 1$. Note that when $\lambda = 1$ the expanded model is reduced to the variance components model given in (8.1).

The PXEM update for $d$ is equivalent to the EM update for $\sigma_u^2$ and is given by

$$
\sigma_u^2(m+1) = \frac{\mathbf{u}^{(m)'} \mathbf{u}^{(m)} + \text{tr}(\mathbf{C}^{Z(m)'} \mathbf{Z}(m))}{b}
$$

The PXEM update for $\sigma^2$ is equivalent to the EM update for $\sigma^2$ and is given by

$$
\sigma^2(m+1) = \frac{\mathbf{e}^{(m)'} \mathbf{e}^{(m)} + \text{tr}(\mathbf{W}'(m) \mathbf{W})}{n}
$$

The PXEM update for $\lambda$ is given by

$$
\lambda(m+1) = \frac{\mathbf{u}^{(m)'} \mathbf{Z}'(\mathbf{y} - \mathbf{X}\hat{\tau}^{(m)}) - \text{tr}(\mathbf{Z}' \mathbf{C}^{XZ(m)})}{\mathbf{u}^{(m)'} \mathbf{Z} \mathbf{u}^{(m)} + \text{tr}(\mathbf{Z}' \mathbf{C}^{Z(m)'} \mathbf{Z}(m))}
$$

Hence the PXEM update for $\sigma_u^2$ is given by

$$
\sigma_u^2(m+1) = \lambda(m+1)^2 \sigma_u^2(m+1) \tag{8.7}
$$

From Section 6.3.5, the convergence rate matrix for the PXEM algorithm is given by

$$
\mathbf{M}_{\text{PX}}' = \mathbf{I} - \mathbf{I}_{\text{Ec}}(\mathbf{K})^{-1} \mathbf{I}_{\mathbf{O}}(\mathbf{K})
$$

where

$$
\mathbf{I}_{\text{Ec}}(\mathbf{K}) = \mathbb{E} \left[ -\frac{\partial^2 \ell_c(\mathbf{K})}{\partial \mathbf{K} \partial \mathbf{K}'} \bigg| \mathbf{y}_2 \right]
$$

Note that $\mathbf{K} = (\mathbf{K}, \lambda) = (\sigma_u^2, \sigma^2, \lambda)$ for the variance components model.

The REML PXEM complete data log-likelihood for the variance components model is given by

$$
\ell_c(\mathbf{K}) = \frac{1}{2} \left( b \log \sigma^2_u + \frac{\mathbf{f}' \mathbf{f}}{\sigma^2_u} \right) - \frac{1}{2} \left( n \log \sigma^2 + \frac{\mathbf{e}' \mathbf{e}}{\sigma^2} \right)
$$

$$
= \frac{1}{2} \left( b \log \sigma^2_u - 2b \log \lambda + \lambda^2 \mathbf{f}' \mathbf{f} \right) - \frac{1}{2} \left( n \log \sigma^2 + \frac{\mathbf{e}' \mathbf{e}}{\sigma^2} \right)
$$

since $\sigma^2_d = \frac{\sigma^2}{\lambda^2}$. Note that $\mathbf{e} = \mathbf{y} - \mathbf{X}\hat{\tau} - \lambda \mathbf{Z} \mathbf{f}$ and hence

$$
\mathbf{e}' \mathbf{e} = (\mathbf{y} - \mathbf{X}\hat{\tau} - \lambda \mathbf{Z} \mathbf{f})' (\mathbf{y} - \mathbf{X}\hat{\tau} - \lambda \mathbf{Z} \mathbf{f})
$$

$$
= (\mathbf{y} - \mathbf{X}\hat{\tau})' (\mathbf{y} - \mathbf{X}\hat{\tau}) - 2\lambda \mathbf{f}' \mathbf{Z}' (\mathbf{y} - \mathbf{X}\hat{\tau}) + \lambda^2 \mathbf{f}' \mathbf{Z}' \mathbf{Z} \mathbf{f}
$$

and

$$
\frac{\partial \mathbf{e}' \mathbf{e}}{\partial \lambda} = -2 \mathbf{f}' \mathbf{Z}' (\mathbf{y} - \mathbf{X}\hat{\tau}) + 2\lambda \mathbf{f}' \mathbf{Z}' \mathbf{Z} \mathbf{f}
$$
The derivatives of $\ell_c(K)$ needed to calculate $\mathcal{I}_{E_c}(K)$ are

$$
\frac{\partial \ell_c(K)}{\partial \sigma_u^2} = -\frac{1}{2} \left( \frac{b}{\sigma_u^4} - \frac{\lambda^2 f' f}{\sigma_u^4} \right)
$$

$$
\frac{\partial^2 \ell_c(K)}{\partial \sigma_u^2^2} = -\frac{1}{2} \left( \frac{b}{\sigma_u^4} + \frac{2\lambda^2 f' f}{\sigma_u^4} \right)
$$

$$
\frac{\partial^2 \ell_c(K)}{\partial \sigma_u^2 \partial \sigma} = \frac{\partial^2 \ell_c(K)}{\partial \sigma^2 \partial \sigma_u^2} = 0
$$

$$
\frac{\partial^2 \ell_c(K)}{\partial \sigma_u^4} = \frac{\partial^2 \ell_c(K)}{\partial \sigma \partial \sigma_u^2} = -\frac{1}{2} \left( \frac{-2\lambda f' f}{\sigma_u^4} \right)
$$

$$
\frac{\partial \ell_c(K)}{\partial \sigma^2} = -\frac{1}{2} \left( \frac{n}{\sigma^4} - \frac{e' e}{\sigma^4} \right)
$$

$$
\frac{\partial^2 \ell_c(K)}{\partial \sigma^2} = -\frac{1}{2} \left( \frac{n}{\sigma^4} + \frac{2e' e}{\sigma^6} \right)
$$

$$
\frac{\partial^2 \ell_c(K)}{\partial \sigma^2 \partial \lambda} = \frac{\partial^2 \ell_c(K)}{\partial \lambda \partial \sigma^2} = -\frac{1}{2} \left( \frac{2f'Z'(y - X\tau) - 2\lambda f'Z'Z f}{\sigma^4} \right)
$$

$$
\frac{\partial \ell_c(K)}{\partial \lambda} = -\frac{1}{2} \left( \frac{-2b}{\lambda^2} + \frac{2\lambda f' f}{\sigma_u^2} \right) - \frac{1}{2} \left( \frac{-2f'Z'(y - X\tau) + 2\lambda f'Z'Z f}{\sigma^2} \right)
$$

$$
\frac{\partial^2 \ell_c(K)}{\partial \lambda^2} = -\frac{1}{2} \left( \frac{2b}{\lambda^2} + \frac{2f' f}{\sigma_u^2} \right) - \frac{1}{2} \left( \frac{2f'Z'Z f}{\sigma^2} \right)
$$

Hence the elements of the REML PXEM conditional expected information matrix for the complete data are

$$
\mathcal{I}_{E_c}(\sigma_u^2, \sigma_u^2) = E \left[ -\frac{\partial^2 \ell_c(K)}{\partial(\sigma_u^2)^2} \mid y_2 \right]
$$

$$
= \frac{1}{2} \left( \frac{b}{\sigma_u^4} + \frac{2\lambda^2 E[f' f | y_2]}{\sigma_u^6} \right)
$$

$$
= \frac{1}{2} \left( \frac{b}{\sigma_u^4} + \frac{2\lambda^2 \left( \hat{f}' \hat{f} + \text{tr}(C_p^Z) \right)}{\sigma_u^6} \right)
$$

$$
\mathcal{I}_{E_c}(\sigma_u^2, \sigma^2) = \mathcal{I}_{E_c}(\sigma_u^2, \sigma_u^2)
$$

$$
= E \left[ \frac{\partial^2 \ell_c(K)}{\partial \sigma_u^2 \partial \sigma} \mid y_2 \right]
$$

$$
= 0
$$

$$
\mathcal{I}_{E_c}(\sigma_u^2, \lambda) = \mathcal{I}_{E_c}(\lambda, \sigma_u^2)
$$

$$
= E \left[ -\frac{\partial^2 \ell_c(K)}{\partial \sigma_u^2 \partial \lambda} \mid y_2 \right]
$$

$$
= \frac{1}{2} \left( \frac{-2\lambda E[f' f | y_2]}{\sigma_u^4} \right)
$$

$$
= \frac{1}{2} \left( \frac{2\lambda \left( \hat{f}' \hat{f} + \text{tr}(C_p^Z) \right)}{\sigma_u^4} \right)
$$
\[ I_{Ec}(\sigma^2, \lambda) = I_{Ec}(\lambda, \sigma^2) \]
\[ = \mathbb{E} \left[ -\frac{\partial^2 \ell_c(K)}{\partial \sigma^2} \bigg| y_2 \right] \]
\[ = \frac{1}{2} \left( \frac{n}{\sigma^4} - \frac{2 \mathbb{E}[e'e|y_2]}{\sigma^6} \right) \]
\[ = \frac{1}{2} \left( \frac{n}{\sigma^4} + 2 (\hat{e}'\hat{e} + \text{tr}(W C^{-1} W')) \right) \]
\[ I_{Ec}(\lambda, \lambda) = I_{Ec}(\hat{\lambda}, \sigma^2) \]
\[ = \mathbb{E} \left[ -\frac{\partial^2 \ell_c(K)}{\partial \lambda^2} \bigg| y_2 \right] \]
\[ = \frac{1}{2} \left( \frac{2b}{\lambda^2} + \frac{2 \mathbb{E}[f'f|y_2]}{\sigma_u^2} \right) + \frac{1}{2} \left( \frac{2 \mathbb{E}[f'f|y_2]}{\sigma^2} \right) \]
\[ = \frac{1}{2} \left( \frac{2b}{\lambda^2} + \frac{2 (\hat{f}'\hat{f} + \text{tr}(C_{ZZ}^p))}{\sigma_u^2} \right) + \frac{1}{2} \left( \frac{2 (\hat{f}'\hat{f} + \text{tr}(C_{ZC}^{ZZ}))}{\sigma^2} \right) \]

Evaluating the elements of \( I_{Ec}(K) \) at \( \hat{K} = (\hat{\kappa}, \lambda_0) \), where \( \lambda_0 = 1 \), gives

\[ I_{Ec}(\sigma_u^2, \sigma_u^2) = \frac{1}{2} \left( -\frac{b}{\sigma_u^4} + \frac{2 (\hat{u}'\hat{u} + \text{tr}(C^{ZZ}))}{\sigma_u^6} \right) \]
\[ = \frac{b}{2\sigma_u^4} \text{ (since } b\sigma_u^2 = \hat{u}'\hat{u} + \text{tr}(C^{ZZ})) \]
\[ I_{Ec}(\sigma_u^2, \sigma^2) = I_{Ec}(\sigma^2, \sigma_u^2) \]
\[ = 0 \]
\[ I_{Ec}(\sigma_u^2, \lambda_0) = I_{Ec}(\lambda_0, \sigma_u^2) \]
\[ = \frac{1}{2} \left( -\frac{2 (\hat{u}'\hat{u} + \text{tr}(C^{ZZ}))}{\sigma_u^4} \right) \]
\[ = -\frac{b}{\sigma_u^4} \]
\[ I_{Ec}(\sigma^2, \sigma^2) = \frac{1}{2} \left( -\frac{n}{\sigma^4} + \frac{2 (\hat{e}'\hat{e} + \text{tr}(W C^{-1} W'))}{\sigma^6} \right) \]
\[ = \frac{n}{2\sigma^4} \text{ (since } n\sigma^2 = \hat{e}'\hat{e} + \text{tr}(W C^{-1} W')) \]
\[ \mathcal{I}_{E_c}(\hat{\sigma}^2, \lambda_0) = \mathcal{I}_{E_c}(\lambda_0, \hat{\sigma}^2) = 0 \text{ (see below)} \]

\[ \mathcal{I}_{E_c}(\lambda_0, \lambda_0) = \frac{1}{2} \left( 2b + \frac{2(\hat{u}'\hat{u} + \text{tr}(C^{ZZ}))}{\hat{\sigma}^2} \right) + \frac{1}{2} \left( \frac{2(\hat{u}'Z'\hat{u} + \text{tr}(Z'ZC^{ZZ}))}{\hat{\sigma}^2} \right) \]

\[ = 2b + \frac{\hat{u}'Z'\hat{u} + \text{tr}(Z'ZC^{ZZ})}{\hat{\sigma}^2} \]

To show that \( \mathcal{I}_{E_c}(\hat{\sigma}^2, \lambda_0) = 0 \) consider \( \mathcal{I}_{E_c}(\sigma^2, \lambda) \) at convergence, that is evaluated at \( \kappa = \hat{\kappa} \) and \( \lambda = \hat{\lambda} \):

\[ \mathcal{I}_{E_c}(\frac{1}{2\sigma^2}) = \frac{1}{2\sigma^2} \left( 2(\hat{u}'Z'(y - X\hat{\tau}) - \text{tr}(Z'XC^{XZ})) - 2\hat{\lambda}(\hat{u}'Z'\hat{u} + \text{tr}(Z'ZC^{ZZ})) \right) \]

since

\[ \hat{\lambda} = \frac{\hat{u}'Z'(y - X\hat{\tau}) - \text{tr}(Z'XC^{XZ})}{\hat{u}'Z'\hat{u} + \text{tr}(Z'ZC^{ZZ})} \]

Therefore \( \mathcal{I}_{E_c}(\hat{\sigma}^2, \lambda_0) = 0 \).

This allows us to write \( \mathcal{I}_{E_c}(\hat{\kappa}) \) as

\[
\mathcal{I}_{E_c}(\hat{\kappa}) = \begin{bmatrix}
\frac{b}{2\sigma^2} & 0 & -\frac{b}{\sigma^2} \\
0 & \frac{n}{2\sigma^2} & 0 \\
-\frac{b}{\sigma^2} & 0 & 2b + \frac{\hat{u}'Z'\hat{u} + \text{tr}(Z'ZC^{ZZ})}{\sigma^2}
\end{bmatrix}
\]

Recall from Section 6.3.5 that the convergence of the PXEM algorithm is determined by the convergence of \( \kappa^{(m)} \), and the rate matrix for \( \kappa \) for the PXEM algorithm is given by

\[ M'_{P_X}(\hat{\kappa}) = I - \mathcal{I}_{O}(\hat{\kappa}) \]

where

\[ \mathcal{I}_{E_c}(\hat{\kappa})^{-1} = \begin{bmatrix}
V_{\hat{\kappa}\hat{\kappa}} & V_{\hat{\kappa}\lambda_0} \\
V_{\lambda_0\hat{\kappa}} & V_{\lambda_0\lambda_0}
\end{bmatrix} \]

Note that when the data are balanced with \( k \) observations for each of \( b \) blocks, \( Z'Z = kI_b \) and hence

\[ \hat{u}'Z'\hat{u} + \text{tr}(Z'ZC^{ZZ}) = k \left( \hat{u}'\hat{u} + \text{tr}(C^{ZZ}) \right) \]

\[ = kb\hat{\sigma}^2 \]

\[ = n\hat{\sigma}^2 \]

therefore

\[ \mathcal{I}_{E_c}(\lambda_0, \lambda_0) = \frac{2b + \frac{n\hat{\sigma}^2}{\sigma^2}}{\sigma^2} \]

\[ = \frac{2b\hat{\sigma}^2 + n\hat{\sigma}^2}{\sigma^2} \]
and

$$\mathcal{I}_{E^c}(\hat{\kappa}) = \begin{bmatrix} \frac{b}{2\sigma_u^4} & 0 & -\frac{b}{\sigma_u^2} \\ 0 & \frac{2n}{\sigma_u^2} & 0 \\ -\frac{b}{\sigma_u^2} & 0 & \frac{2b\sigma_u^2 + n\sigma_u^4}{\sigma^2} \end{bmatrix}$$

Using Result A.18

$$V_{\hat{\kappa}\hat{\kappa}} = \begin{bmatrix} 4\hat{\sigma}_u^2 & 0 \\ 0 & \frac{2\sigma_u^4}{n} \end{bmatrix}$$

Therefore the convergence rate matrix is given by

$$M'_{PX}(\hat{\kappa}) = I - V_{\hat{\kappa}\hat{\kappa}} \mathcal{I}_O(\hat{\kappa})$$

$$= I - \begin{bmatrix} \frac{4\sigma_u^2 + 2k\sigma_u^4}{n} & 0 \\ 0 & \frac{2\sigma_u^4}{n} \end{bmatrix} \begin{bmatrix} \mathcal{I}_O(\hat{\sigma}_u^2, \hat{\sigma}_u^2) & \mathcal{I}_O(\hat{\sigma}_u^2, \hat{\sigma}_u^2) & \mathcal{I}_O(\hat{\sigma}_u^2, \hat{\sigma}_u^2) \\ \mathcal{I}_O(\hat{\sigma}_u^2, \hat{\sigma}_u^2) & \mathcal{I}_O(\hat{\sigma}_u^2, \hat{\sigma}_u^2) & \mathcal{I}_O(\hat{\sigma}_u^2, \hat{\sigma}_u^2) \end{bmatrix}$$

and the rate of convergence is given by the largest eigenvalue of $M'_{PX}(\hat{\kappa})$.

### 8.2.4 Local EM Scheme

The local data is defined as

$$v = \Omega^{x(m)} Z' S^{x(m)} y$$

where

$$\Omega^{x(m)} = (C^{ZZ^{x(m)}})^{-1} - I_b/\sigma_u^{2(m)}$$

and $C^{ZZ^{x(m)}}$ is an approximation of $C^{ZZ^{x(m)}}$ where all off diagonal elements are set to zero, that is $C^{ZZ^{x(m)}} = \text{diag}(C^{ZZ^{x(m)}})$.

The local data has distribution $v \sim N(\mathbf{0}, \sigma_u^2 I_b + \Omega^{x(m)^{-1}})$ and the local variance components model is given by

$$v = u_v + e_v \quad (8.8)$$

where

$$\begin{bmatrix} u_v \\ e_v \end{bmatrix} \sim N\left( \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma_u^2 I_b & 0 \\ 0 & (\Omega^{x(m)^{-1}}) \end{bmatrix} \right)$$

The local M-step for $\sigma_u^2$ is given by

$$\frac{\partial E_{EM}(l_v(u_v))^{(n)}}{\partial \sigma_u^2} = -\frac{1}{2} \left( \frac{b}{\sigma_u^2} - \frac{\text{tr}(C_v^{-1(n)})}{\sigma_u^4} - \frac{u_v(n)' u_v(n)}{\sigma_u^4} \right)$$
Equating to zero and solving for \( \sigma^2_u \) we obtain the local EM update for \( \sigma^2_u \),

\[
\sigma^2_u(n+1) = \frac{\tilde{u}^{(n)'} \tilde{u}^{(n)} + \text{tr}(C_v^{-1(n)})}{b}
\]

where \( C_v^{(n)} = \Omega^{*(m)} + I_b/\sigma^2_u \) and \( \tilde{u}^{(n)} = C_v^{-1(n)} \Omega^{*(m)} v \).

Note that on the first iteration of the local EM iteration \( C_v^{(n)} = (C_{ZZ}^{*(m)})^{-1} \) and \( \tilde{u}^{(n)} = \tilde{u}^{(m)} \). And since \( \text{tr}(C_{ZZ}^{*(m)}) = \text{tr}(C_{ZZ}^{*(m)}) \), the local EM update for \( \sigma^2_u \) is given by

\[
\sigma^2_u(n+1) = \frac{\tilde{u}^{(m)'} \tilde{u}^{(m)} + \text{tr}(C_{ZZ}^{*(m)})}{b}
\]

which is the same as the EM update for \( \sigma^2_u \) as given in (8.5). Hence the update for \( \sigma^2_u \) from the first iteration of the local EM scheme is equivalent to the update for \( \sigma^2_u \) from a full EM iteration.

### 8.2.5 Local PXEM Scheme

The local model in (9.2.5) is expanded to

\[
v = \lambda f_v + e_v
\]

such that \( u_v = \lambda f_v \) where \( f \) is a vector of rescaled random effects. The joint distribution of \( f_v \) and \( e_v \) is given by

\[
\begin{bmatrix} f_v \\ e_v \end{bmatrix} \sim N \left( \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma^2_d I_b & 0 \\ 0 & \Omega^{*(m)}(-1) \end{bmatrix} \right)
\]

Note that

\[
\begin{align*}
\text{Var}(u_v) &= \lambda^2 \text{Var}(f_v) \\
\sigma^2_u I_b &= \lambda^2 \sigma^2_d I_b \\
\sigma^2_u &= \lambda^2 \sigma^2_d
\end{align*}
\]

The local PXEM update for \( \sigma^2_d \) is equivalent to the local EM update for \( \sigma^2_u \) and is given by

\[
\sigma^2_d(n+1) = \frac{\tilde{u}^{(n)'} \tilde{u}^{(n)} + \text{tr}(C_v^{-1(n)})}{b}
\]

The local PXEM update for \( \lambda \) is given by

\[
\lambda^{(n+1)} = \frac{\tilde{u}^{(n)'} \Omega^{*(m)} v}{\tilde{u}^{(n)'} \Omega^{*(m)} \tilde{u}^{(m)} + \text{tr}(\Omega^{*(m)} C_v^{-1(n)})}
\]

Hence the local PXEM update for \( \sigma^2_u \) is given by

\[
\sigma^2_u^{(n+1)} = \lambda^{(n+1)^2} \sigma^2_d^{(n+1)}
\]
8.2.6 Starting Values

Starting values for $\sigma^2_u$ and $\sigma^2$ can be calculated using Henderson’s Method III. Recall from Section 7.4 that Henderson’s Method III estimates variance components using the sums of squares that result from fitting a linear model and various submodels of it. The sums of squares are computed as though the random effects are fixed effects and are then equated to their expected values which are derived under the mixed model. The estimates of the variance parameters are obtained by solving for these equations for the variance parameters.

We write the variance components model in (8.1) as

$$y = X\tau + Zu + e = W\beta + e$$

where $\beta = \begin{bmatrix} \tau \\ u \end{bmatrix}$ and $W = \begin{bmatrix} X \\ Z \end{bmatrix}$, to remove the distinction between fixed and random effects. In fitting a fixed effects model of the form $y = W\beta + e$ where $e \sim N(0, \sigma^2 I_n)$, the best linear unbiased estimator of $\beta$ is given by

$$\hat{\beta} = (W'W)^{-}W'y$$

where $(W'W)^{-}$ is the generalised inverse, with $(W'W)^{-} = (W'W)^{-1}$ when $W'W$ has full rank.

The first sums of squares that we consider is the error sums of squares from fitting the model $y = W\beta + e$.

$$SSE = \hat{e}'\hat{e} = (y - W\hat{\beta})'(y - W\hat{\beta}) = y'y - y'W\hat{\beta} - \hat{\beta}'W'y + \hat{\beta}'W'W\hat{\beta} = y'y - y'W(W'W)^{-}W'y$$

The reduction in sums of squares due to fitting $y = W\beta + e$ is

$$R(\beta) = y'W(W'W)^{-}W'y$$

Note that $R(\beta) = R(\tau, u)$.

We then consider the submodel $y = X\beta + e$. The reduction in sums of squares due to fitting this model is

$$R(\tau) = y'X(X'X)^{-1}X'y$$

The reduction due to fitting $u$ after accounting for $\tau$ is

$$R(u|\tau) = R(\tau, u) - R(\tau) = y'W(W'W)^{-}W'y - y'X(X'X)^{-1}X'y$$
8.2. VARIANCE COMPONENTS MODEL AND RESULTS

Henderson’s Method III estimates the variance components by equating the computed reduction in sums of squares to its expected value under the full model. Hence we need to find the expected values of

\[
SSE = y'y - R(\tau, u) \\
= y'y - y'W(W'W)^{-1}W'y
\]

\[
R(u|\tau) = R(\tau, u) - R(\tau) \\
= y'W(W'W)^{-1}W'y - y'X(X'X)^{-1}X'y
\]

Note the general result from Result A.10,

\[
E[y'AY] = \text{tr}(A \text{Var}[y]) + E[y']AE[y]
\]

Here \( y = W\beta + e \), therefore

\[
E(y) = WE[\beta] + E[e] \\
= WE[\beta]
\]

and

\[
\text{Var}[y] = W\text{Var}[\beta]W' + \text{Var}[e] \\
= W\text{Var}[\beta]W' + \sigma^2 I_n.
\]

It follows then

\[
E[y'AY] = \text{tr}(A(W\text{Var}[\beta]W' + \sigma^2 I_n)) + E[y']AE[y]
\]

\[
= \text{tr}(A\text{Var}[\beta]W') + \sigma^2\text{tr}(A) + \text{tr}(E[\beta']W'AWE[\beta])
\]

\[
= \text{tr}(W'AW\text{Var}[\beta]) + \text{tr}(W'AWE[\beta]E[\beta']) + \sigma^2\text{tr}(A)
\]

\[
= \text{tr}(W'AWE[\beta']) + \sigma^2\text{tr}(A)
\]

since \( E[\beta\beta'] = \text{Var}[\beta] + E[\beta]E[\beta'] \).

Hence

\[
E[y'y] = \text{tr}(W'WE[\beta\beta']) + n\sigma^2
\]

\[
E[y'W(W'W)^{-1}W'y] = \text{tr}(W'W(W'W)^{-1}W'WE[\beta\beta']) + \sigma^2\text{tr}(W(W'W)^{-1}W')
\]

\[
= \text{tr}(W'WE[\beta\beta']) + \sigma^2\text{tr}(W(W'W)^{-1}W')
\]

, and

\[
E[y'X(X'X)^{-1}X'y] = \text{tr}(W'X(X'X)^{-1}X'WE[\beta\beta']) + \sigma^2\text{tr}(X(X'X)^{-1}X')
\]
Therefore
\[
E[SSE] = E[y'y] - E[y'W(W'W)^{-1}W'y]
\]
\[
= \text{tr}(W'WE[\beta\beta']) + n\sigma^2 - \text{tr}(W'WE[\beta\beta']) - \sigma^2\text{tr}(W(W'W)^{-1}W')
\]
\[
= n\sigma^2 - \sigma^2\text{tr}(W(W'W)^{-1}W')
\]
\[
= \sigma^2(n - r(W)) \quad \text{Using Result A.15}
\]
and
\[
E[R(u|\tau)] = E[y'W(W'W)^{-1}W'y] - E[y'X(X'X)^{-1}X'y]
\]
\[
= \text{tr}(W'WE[\beta\beta']) + \sigma^2\text{tr}(W(W'W)^{-1}W')
\]
\[
- \text{tr}(W'X(X'X)^{-1}X'WE[\beta\beta']) - \sigma^2\text{tr}(X(X'X)^{-1}X')
\]
\[
= \text{tr}(W'(I - X(X'X)^{-1}X')WE[\beta\beta'])
\]
\[
+ \sigma^2(\text{tr}(W(W'W)^{-1}W') - \text{tr}(X(X'X)^{-1}X'))
\]
\[
= \text{tr}(W'M_XWE[\beta\beta']) + \sigma^2(\text{tr}(W(W'W)^{-1}W') - \text{tr}(X(X'X)^{-1}X'))
\]
\[
= \text{tr}(W'M_XWE[\beta\beta']) + \sigma^2(r(W) - r(X)) \quad \text{Using Result A.15}
\]
where \(M_X = I - X(X'X)^{-1}X'\).

The expectation is calculated under the full model \(y = X\tau + Zu = e\) where \(\tau\) is a vector of fixed effects and hence \(E[\tau] = \tau\) and \(\text{Var}[\tau] = 0\), and \(u\) is a vector of random effects with \(E[u] = 0\) and \(\text{Var}[u] = \sigma_u^2I_b\). Therefore
\[
E[\beta\beta'] = \begin{bmatrix}
E[\tau\tau'] & E[\tau'u'] \\
E[u\tau'] & E[uu']
\end{bmatrix}
\]
\[
= \begin{bmatrix}
\tau\tau' & 0 \\
0 & \sigma_u^2I_b
\end{bmatrix}
\]

since \(E[uu'] = \text{Var}[u] + E[u]E[u'] = \text{Var}[u] = \sigma_u^2I_b\).

Consider
\[
W'M_XW = \begin{bmatrix}
X' \\
Z'
\end{bmatrix} M_X \begin{bmatrix}
X \\
Z
\end{bmatrix}
\]
\[
= X'M_XX \\
Z'M_XX \\
X'M_XZ \\
Z'M_XZ
\]
\[
= \begin{bmatrix}
0 \\
X'M_XZ \\
Z'M_XX \\
Z'M_XZ
\end{bmatrix}
\]

since
\[
X'M_XX = X'(I - X(X'X)^{-1}X')X
\]
\[
= X'X - X'X(X'X)^{-1}X'X
\]
\[
= 0
\]

Thus
\[
W'M_XWE[\beta\beta'] = \begin{bmatrix}
0 \\
Z'M_XX \\
Z'M_XZ \\
Z'M_XX\tau\tau'
\end{bmatrix} \begin{bmatrix}
\tau\tau' & 0 \\
0 & \sigma_u^2I_b
\end{bmatrix}
\]
\[
= \begin{bmatrix}
0 \\
\sigma_u^2X'M_XZ \\
\sigma_u^2Z'M_XZ
\end{bmatrix}
\]
Therefore
\[
\text{tr}(W'M_XW)E[\beta\beta'] = \sigma_u^2\text{tr}(Z'M_XZ)
\]
and hence
\[
E[R(u|\tau)] = \sigma_u^2\text{tr}(Z'M_XZ) + \sigma^2(r(W) - r(X))
\]

This allows us to write the expected values of the sums of squares of interest as
\[
E[SSE] = \sigma^2(n - r(W))
\]
\[
E[R(u|\tau)] = \sigma_u^2\text{tr}(Z'M_XZ) + \sigma^2(r(W) - r(X))
\]

Estimates of the variance parameters are obtained by equating the calculated sums of squares to their expected values and then solving for the variance parameters.

An estimate of \(\sigma^2\) is obtained by equating the calculated sums of squares \(SSE\) to their expected value and solving for \(\sigma^2\). This gives
\[
\hat{\sigma}^2 = \frac{SSE}{(n - r(W))}
\]
where \(SSE = y'y - y'W(W'W)^{-1}W'y\) and \(W = [X \; Z]\).

An estimate of \(\sigma_u^2\) is obtained by equating the calculated sums of squares \(R(u|\tau)\) to their expected value and solving for \(\sigma_u^2\). This gives
\[
\hat{\sigma}_u^2 = \frac{R(u|\tau) - \sigma^2(r(W) - r(X))}{\text{tr}(Z'M_XZ)}
\]
where \(R(u|\tau) = y'W(W'W)^{-1}W'y - y'X(X'X)^{-1}X'y\).

8.3 Lamb Weight Data

We consider the lamb weight data analysed by Harville and Fenech (1985), Harville and Callanan (1991) and Callanan and Harville (1991). The data consists of the birth weights (in pounds) of 62 single-birth male lambs. The data come from 5 distinct population lines. Each lamb was the progeny of 1 of 23 rams. Each lamb had a different dam. Age of dam was recorded as belonging to one of three categories (1-2 years, 2-3 years and over 3 years). The data are presented in Table 8.1.

We consider the model fitted by Harville and Fenech (1985) in which \(y_{ijkl}\) represents the birth weight of the \(l\)th lamb that is the offspring of the \(k\)th sire from the \(j\)th population line and a dam belonging to the \(i\)th age category, and follows the variance components model
\[
y_{ijkl} = \alpha_i + \beta_j + \gamma_k + \epsilon_{ijkl}
\]
Table 8.1: The lamb weight data with the birth weights given in pounds.

<table>
<thead>
<tr>
<th>Line</th>
<th>Sire</th>
<th>Dam age</th>
<th>Weight</th>
<th>Line</th>
<th>Sire</th>
<th>Dam age</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>6.2</td>
<td>4</td>
<td>13</td>
<td>1</td>
<td>9.2</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td>13.0</td>
<td>4</td>
<td>13</td>
<td>1</td>
<td>10.6</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
<td>9.5</td>
<td>4</td>
<td>13</td>
<td>1</td>
<td>10.6</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
<td>10.1</td>
<td>4</td>
<td>13</td>
<td>3</td>
<td>7.7</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
<td>11.4</td>
<td>4</td>
<td>13</td>
<td>3</td>
<td>10.0</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>2</td>
<td>11.8</td>
<td>4</td>
<td>13</td>
<td>3</td>
<td>11.2</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>3</td>
<td>12.9</td>
<td>4</td>
<td>14</td>
<td>1</td>
<td>10.2</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>3</td>
<td>13.1</td>
<td>4</td>
<td>14</td>
<td>1</td>
<td>10.9</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>1</td>
<td>10.4</td>
<td>4</td>
<td>15</td>
<td>1</td>
<td>11.7</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>2</td>
<td>8.5</td>
<td>4</td>
<td>15</td>
<td>3</td>
<td>9.9</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>3</td>
<td>13.5</td>
<td>5</td>
<td>16</td>
<td>1</td>
<td>11.7</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>2</td>
<td>10.1</td>
<td>5</td>
<td>16</td>
<td>1</td>
<td>12.6</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>3</td>
<td>11.0</td>
<td>5</td>
<td>17</td>
<td>1</td>
<td>9.0</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>3</td>
<td>14.0</td>
<td>5</td>
<td>17</td>
<td>3</td>
<td>11.0</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>3</td>
<td>15.5</td>
<td>5</td>
<td>18</td>
<td>3</td>
<td>9.0</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
<td>1</td>
<td>12.0</td>
<td>5</td>
<td>18</td>
<td>3</td>
<td>12.0</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>1</td>
<td>11.5</td>
<td>5</td>
<td>19</td>
<td>3</td>
<td>9.9</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>3</td>
<td>10.8</td>
<td>5</td>
<td>20</td>
<td>2</td>
<td>13.5</td>
</tr>
<tr>
<td>3</td>
<td>9</td>
<td>2</td>
<td>9.0</td>
<td>5</td>
<td>21</td>
<td>2</td>
<td>10.9</td>
</tr>
<tr>
<td>3</td>
<td>9</td>
<td>3</td>
<td>9.5</td>
<td>5</td>
<td>21</td>
<td>3</td>
<td>5.9</td>
</tr>
<tr>
<td>3</td>
<td>9</td>
<td>3</td>
<td>12.6</td>
<td>5</td>
<td>22</td>
<td>2</td>
<td>10.0</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>1</td>
<td>11.0</td>
<td>5</td>
<td>22</td>
<td>2</td>
<td>12.7</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>2</td>
<td>10.1</td>
<td>5</td>
<td>22</td>
<td>3</td>
<td>13.2</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>2</td>
<td>11.7</td>
<td>5</td>
<td>22</td>
<td>3</td>
<td>13.3</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>3</td>
<td>8.5</td>
<td>5</td>
<td>23</td>
<td>1</td>
<td>10.7</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>3</td>
<td>8.8</td>
<td>5</td>
<td>23</td>
<td>1</td>
<td>11.0</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>3</td>
<td>9.9</td>
<td>5</td>
<td>23</td>
<td>1</td>
<td>12.5</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>3</td>
<td>10.9</td>
<td>5</td>
<td>23</td>
<td>3</td>
<td>9.0</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>3</td>
<td>11.0</td>
<td>5</td>
<td>23</td>
<td>3</td>
<td>10.2</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>1</td>
<td>11.6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>3</td>
<td>13.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>12</td>
<td>2</td>
<td>12.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
where $\alpha_i$ are the fixed age effects $i = 1, 2, 3$, $\beta_j$ are the fixed line effects $j = 1, 2, 3, 4, 5$ and $\gamma_k$ are the random sire effects $k = 1, \ldots, 23$.

In matrix notation we have

$$
y = X\tau + Zu + e$$

where $y$ is the $62 \times 1$ vector of birth weights, $\tau$ is the $7 \times 1$ vector of fixed effects (comprising of an intercept and age and line effects) and $X$ is the associated $62 \times 7$ design matrix of full column rank, $u$ is the $23 \times 1$ vector of random sire effects, $Z$ is the associated $62 \times 23$ design matrix, and $e$ is the vector of residuals. It is assumed that the joint distribution of $u$ and $e$ is given by

$$
\begin{bmatrix} u \\ e \end{bmatrix} \sim N \left( \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma_u^2 I_{23} & 0 \\ 0 & \sigma^2 I_{62} \end{bmatrix} \right)
$$

where $\sigma_u^2$ is the variance of the random effects and $\sigma^2$ is the variance of the residuals.

We used the AI, EM and PXEM algorithms to obtain the REML estimates of the variance components $\sigma_u^2$ and $\sigma^2$. We used the convergence criterion defined in 6.1 with $\kappa' = (\sigma_u^2, \sigma^2)$. The analyses were conducted in the statistical packages S-PLUS (MathSoft, Inc, 1999) and R (R Development Core Team, 2007), using a suite of purpose-built functions. The analyses were conducted using two different sets of starting values; uninformed starting values of $\sigma_u^2(0) = 1$ and $\sigma^2(0) = 1$, and informed starting values of $\sigma_u^2(0) = 0.77$ and $\sigma^2(0) = 2.76$, obtained by means of Henderson’s Method III. The REML estimates of the variance components are $\hat{\sigma}_u^2 = 0.52$ and $\hat{\sigma}^2 = 2.96$. These values are presented in Table 8.2.

Table 8.2: The uninformed and informed starting values and the REML estimates for $\sigma_u^2$ and $\sigma^2$ for the lamb weight data analysis.

<table>
<thead>
<tr>
<th>Starting values</th>
<th>REML estimates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uninformed</td>
<td>Informed</td>
</tr>
<tr>
<td>$\sigma_u^2$</td>
<td>1.00</td>
</tr>
<tr>
<td>$\sigma^2$</td>
<td>1.00</td>
</tr>
</tbody>
</table>

The number of iterations to convergence for each algorithm and each set of starting values is given in Table 8.3, and the iteration sequences are shown in Figure 8.1. Looking at the results in Table 8.3 obtained using the uninformed starting values, the AI algorithm converged in the fewest iterations, followed by the PXEM algorithm and then the EM algorithm. The corresponding plots of the iteration sequence for $\sigma_u^2$ and $\sigma^2$ in Figure 8.1 show that the EM and PXEM estimates of the variance parameters rapidly approached the REML estimates in the early iterations, but then took a large number of iterations to converge, particularly the EM algorithm.

Figure 8.2 shows the iteration sequences over the first 20 iterations to allow us to better understand the iteration behaviour in the early iterations. Again, focusing on the results obtained using the uninformed starting values, we see that the first two iterations of the
Table 8.3: The number of iterations to convergence for the various algorithms for the lamb weight data using the uninformed and informed starting values.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Uninformed</th>
<th>Informed</th>
</tr>
</thead>
<tbody>
<tr>
<td>AI</td>
<td>13</td>
<td>10</td>
</tr>
<tr>
<td>EM</td>
<td>337</td>
<td>324</td>
</tr>
<tr>
<td>PXEM</td>
<td>76</td>
<td>71</td>
</tr>
</tbody>
</table>

Figure 8.1: Iteration sequence for $\sigma^2_u$ and $\sigma^2$ for the AI, EM and PXEM algorithms for the lamb weight data with uninformed and informed starting values.

EM and PXEM iterations actually took $\sigma^2_u$ closer to the REML estimate than the first two iterations of the AI algorithm. However, from this point on the AI estimates of $\sigma^2_u$ were closer to the REML estimate than the EM and PXEM iterations. Looking at the iteration sequences for $\sigma^2$, the AI estimates were closer to the REML estimate for all iterations. The PXEM estimates were close to the REML solution within 20 iterations, however the EM estimates took over 70 iterations to get as close to the REML solution, at which stage the PXEM algorithm had almost converged.

The results obtained using the informed starting values were similar to those obtained using the uninformed starting values, with each of the algorithms converging in slightly fewer iterations due to the starting values being closer to the REML estimates. The AI
8.3. *LAMB WEIGHT DATA*

Figure 8.2: Iteration sequence over the first 20 iterations for $\sigma^2_u$ and $\sigma^2$ for the AI, EM and PXEM algorithms for the lamb weight data with uninformed and informed starting values.

The algorithm converged in noticeably fewer iterations than the EM and PXEM algorithm. Looking at the early iterations in Figure 8.2, the AI estimates were closer to the REML estimates at each iteration for both $\sigma^2_u$ and $\sigma^2$. The PXEM estimates were close to the REML estimates within 20 iterations but the EM estimates took over 70 iterations to get as close. It is interesting to note that the use of informed starting values had a minimal impact on the number of iterations to convergence for the three algorithms in this case.

The theoretical and empirical convergence rates for the AI, EM and PXEM algorithms are presented in Table 8.4. The theoretical convergence rate was determined by calculating the largest eigenvalue of the convergence rate matrix for each algorithm as outlined in Section 6.3.6. The empirical convergence rate was calculated using

$$ r_e = \lim_{m \to \infty} \sqrt{\frac{\sum_{i=1}^{k} (\kappa_{i}^{(m+1)} - \kappa_{i}^{(m)})^2}{\sum_{i=1}^{k} (\kappa_{i}^{(m)} - \kappa_{i}^{(m-1)})^2}} $$

as given in (6.17) based on the iteration sequence obtained using the uninformed starting values. The same values were obtained from the iteration sequence obtained using the informed starting values. The empirical convergence rates matched the theoretical convergence rates for all three algorithms, confirming that the algorithms converged.
Table 8.4: The convergence rate for the various algorithms for the lamb weight data.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Theoretical</th>
<th>Empirical</th>
</tr>
</thead>
<tbody>
<tr>
<td>AI</td>
<td>0.17</td>
<td>0.17</td>
</tr>
<tr>
<td>EM</td>
<td>0.96</td>
<td>0.96</td>
</tr>
<tr>
<td>PXEM</td>
<td>0.82</td>
<td>0.82</td>
</tr>
</tbody>
</table>

Figure 8.3 displays a plot of the iteration sequence of

\[ r_e^{(m)} = \sqrt{\frac{\sum_{i=1}^{k} (\kappa^{(m)}_i - \kappa^{(m-1)}_i)^2}{\sum_{i=1}^{k} (\kappa^{(m-1)}_i - \kappa^{(m-2)}_i)^2}} \]

for each of the AI, EM and PXEM algorithms. The fourth plot shows the iteration sequence of \( r_e^{(m)} \) for all three algorithms over the first 10 iterations. The first three plots show \( r_e^{(m)} \) approaching the theoretical convergence rates as the iteration number increases for each of the algorithms. For the AI algorithm, \( r_e^{(m)} \) rapidly decreased as it approached the theoretical convergence rate of 0.17, demonstrating that the AI algorithm rapidly converged in the neighbourhood of the solution. We see the opposite trend for the EM and PXEM algorithms, with \( r_e^{(m)} \) increasing with each iteration as it approached the theoretical convergence rate of 0.96 for the EM algorithm and 0.82 for the PXEM algorithm, indicating that convergence slowed down as the variance parameter estimates approached the REML solution. The fourth plot shows that \( r_e^{(m)} \) for the EM and PXEM algorithms was lower than that of the AI algorithm over the first few iterations, indicating that the EM and PXEM updates were competitive in the early iterations.

These results support the use of a hybrid scheme that uses an EM type algorithm initially and then switches to AI iterations once the variance parameter estimates are within the neighbourhood of the REML solution. However, the AI algorithm worked for both sets of starting values and so there is no need to use the hybrid schemes here. The use of hybrid schemes will be investigated in the following sections using simulated incomplete block design data.

In summary, for the analysis of the lamb weight data, it is evident that the AI algorithm is the preferred algorithm compared to the EM and PXEM algorithms. And although the PXEM algorithm converged in far fewer iterations than the EM algorithm, it fell short of the AI algorithm.
Figure 8.3: Theoretical and empirical convergence rates for the AI, EM and PXEM algorithms for the analysis of the lamb weight data using uninformed starting values. For each algorithm, the dots represent $r_{c}^{(m)}$ at each iteration, and the horizontal line represents the theoretical convergence rate.
8.4 Simulated Incomplete Block Design Data

Data were simulated for incomplete block designs with \( r \) replicates of each of \( t \) treatments organised into \( b \) blocks with \( k \) observations per block. The total number of observations is given by \( n = rt = kb \). If \( y \) is the \( n \times 1 \) vector of observations (in block order) then the model is given by

\[
y = X\tau + Zu + e
\]

where \( \tau \) is the \( t \times 1 \) vector of fixed treatment effects and \( X \) is the associated \( n \times t \) design matrix of full column rank, \( u \) is the \( b \times 1 \) vector of random block effects and \( Z = (I_b \otimes 1_k) \) is the associated \( n \times b \) design matrix, and \( e \) is the \( n \times 1 \) vector of random errors. We assume that the joint distribution of \( u \) and \( e \) is given by

\[
\begin{pmatrix}
u \\ e
\end{pmatrix} \sim N \left( \begin{pmatrix} 0 \\ 0 \\end{pmatrix}, \begin{pmatrix} \sigma_u^2 I_b & 0 \\ 0 & \sigma_e^2 I_n \end{pmatrix} \right)
\]

Two different incomplete block designs were used. The first design had 12 treatments with 9 replicates of each treatment, organised into 12 blocks with 9 observations per block as shown in Table 8.5. The second design had 12 treatments with 3 replicates of each treatment, organised into 12 blocks with 3 observations per block as shown in Table 8.6. So the difference between the designs is the amount of replication and the block size; the first design has 9 replicates of each treatment and 9 observations per block, while the second design only has 3 replicates of each treatment and 3 observations per block. Both designs are partially balanced incomplete block designs and were generated using CycDesigN (Whitaker et al., 2001). For both designs, 20 data sets were simulated using values of \( \sigma_u^2 = 1 \) and \( \sigma_e^2 = 1 \) for the variance parameters.

Table 8.5: Incomplete Block Design with \( t = 12, r = 9, b = 12 \) and \( k = 9 \).

<table>
<thead>
<tr>
<th>Observation</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Block 1</td>
<td>9</td>
<td>4</td>
<td>6</td>
<td>8</td>
<td>11</td>
<td>2</td>
<td>10</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>Block 2</td>
<td>1</td>
<td>6</td>
<td>8</td>
<td>5</td>
<td>10</td>
<td>4</td>
<td>12</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>Block 3</td>
<td>8</td>
<td>3</td>
<td>11</td>
<td>7</td>
<td>2</td>
<td>6</td>
<td>1</td>
<td>12</td>
<td>4</td>
</tr>
<tr>
<td>Block 4</td>
<td>5</td>
<td>1</td>
<td>3</td>
<td>7</td>
<td>9</td>
<td>12</td>
<td>2</td>
<td>8</td>
<td>10</td>
</tr>
<tr>
<td>Block 5</td>
<td>6</td>
<td>12</td>
<td>3</td>
<td>10</td>
<td>9</td>
<td>7</td>
<td>11</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Block 6</td>
<td>5</td>
<td>1</td>
<td>3</td>
<td>7</td>
<td>9</td>
<td>12</td>
<td>2</td>
<td>8</td>
<td>10</td>
</tr>
<tr>
<td>Block 7</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>10</td>
<td>12</td>
<td>2</td>
<td>9</td>
<td>7</td>
<td>11</td>
</tr>
<tr>
<td>Block 8</td>
<td>12</td>
<td>3</td>
<td>8</td>
<td>6</td>
<td>10</td>
<td>11</td>
<td>2</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>Block 9</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>6</td>
<td>9</td>
<td>12</td>
<td>7</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Block 10</td>
<td>11</td>
<td>5</td>
<td>9</td>
<td>7</td>
<td>4</td>
<td>10</td>
<td>12</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>Block 11</td>
<td>1</td>
<td>12</td>
<td>6</td>
<td>10</td>
<td>9</td>
<td>8</td>
<td>11</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Block 12</td>
<td>7</td>
<td>9</td>
<td>1</td>
<td>5</td>
<td>8</td>
<td>2</td>
<td>4</td>
<td>11</td>
<td>6</td>
</tr>
</tbody>
</table>
8.4. SIMULATED INCOMPLETE BLOCK DESIGN DATA

Table 8.6: Incomplete Block Design with $t = 12$, $r = 3$, $b = 12$ and $k = 3$.

<table>
<thead>
<tr>
<th>Observation</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Block</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>12</td>
<td>11</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>12</td>
<td>10</td>
<td>9</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>8</td>
<td>11</td>
</tr>
<tr>
<td>9</td>
<td>6</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>12</td>
<td>7</td>
</tr>
<tr>
<td>11</td>
<td>2</td>
<td>11</td>
<td>9</td>
</tr>
<tr>
<td>12</td>
<td>4</td>
<td>8</td>
<td>1</td>
</tr>
</tbody>
</table>

8.4.1 Incomplete Block Design with 9 replicates

First we will look at the results of the analyses of the 20 data sets with the design shown in Table 8.5, that is the incomplete block design with 9 replicates. The first set of analyses were performed using $\sigma_u^2(0) = 1$ and $\sigma_v^2(0) = 1$ as starting values. The number of iterations to convergence for the AI, EM and PXEM algorithms, as well as the REML estimates of $\sigma_u^2$ and $\sigma_v^2$, for each of the 20 data sets are given in Table 8.7. The iteration sequences for $\sigma_u^2$ and $\sigma_v^2$ for each of the algorithms for each of the data sets are given in Figures 8.4 and 8.5 respectively.

The AI algorithm converged for 19 of the 20 data sets. As expected, the EM and PXEM algorithms converged for all data sets. The number of iterations to convergence for the EM and PXEM algorithms were similar, with the EM algorithm taking the same or slightly more iterations than the PXEM algorithm. For the data sets where AI converged, the AI algorithm converged in fewer iterations than the EM and PXEM algorithms. For the majority of the data sets, the number of iterations to convergence for the AI algorithm was about half the number of iterations to convergence for the PXEM algorithm and the EM algorithm.

The plots in Figures 8.4 and 8.5 show the EM and PXEM updates of $\sigma_u^2$ and $\sigma_v^2$ rapidly approaching the REML solutions over the first few iterations, but then taking at least another 10 iterations to converge. For most of the data sets, the AI updates were not as close to the REML solution as the EM and PXEM updates over the first few iterations. Once the AI updates were close to the REML solution, the AI algorithm converged rapidly. Whereas once the EM and PXEM updates were close to the REML solution, the algorithms were much slower to converge. The PXEM updates for $\sigma_u^2$ approached the REML estimate more rapidly than the EM updates in the early iterations, but there was little difference for $\sigma_v^2$. The PXEM algorithm for the variance components model involves a rescaling of the random effects, and so we expect to see a bigger difference in the behaviour of the EM and PXEM iterations for $\sigma_u^2$ than for $\sigma_v^2$. 
Figure 8.4: Iteration sequence for $\sigma_u^2$ for the AI, EM and PXEM algorithms for the 20 incomplete block design data sets with 9 replicates using $\sigma_u^{2(0)} = 1$ and $\sigma^2(0) = 1$ as starting values.
Figure 8.5: Iteration sequence for $\sigma^2$ for the AI, EM and PXEM algorithms for the 20 incomplete block design data sets with 9 replicates using $\sigma^2_u(0) = 1$ and $\sigma^2(0) = 1$ as starting values.
Table 8.7: The number of iterations to convergence for the AI, EM and PXEM algorithms for the 20 incomplete block design data sets with 9 replicates using $\sigma_u^{2(0)} = 1$ and $\sigma^2(0) = 1$ as starting values.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Algorithm</th>
<th>REML estimates</th>
<th>$\hat{\sigma}_u^2$</th>
<th>$\hat{\sigma}^2$</th>
<th>$\hat{\sigma}_u^2/\hat{\sigma}^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AI 6</td>
<td>EM 13 PXEM 12</td>
<td>1.41</td>
<td>1.13</td>
<td>1.25</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>16</td>
<td>0.56</td>
<td>0.79</td>
<td>0.71</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>13</td>
<td>1.73</td>
<td>0.88</td>
<td>1.96</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td>17</td>
<td>0.54</td>
<td>0.87</td>
<td>0.62</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
<td>21</td>
<td>0.48</td>
<td>1.06</td>
<td>0.45</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>13</td>
<td>1.07</td>
<td>0.86</td>
<td>1.25</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>13</td>
<td>1.16</td>
<td>0.66</td>
<td>1.74</td>
</tr>
<tr>
<td>8</td>
<td>fail</td>
<td>27</td>
<td>0.27</td>
<td>0.91</td>
<td>0.30</td>
</tr>
<tr>
<td>9</td>
<td>5</td>
<td>13</td>
<td>1.26</td>
<td>0.98</td>
<td>1.28</td>
</tr>
<tr>
<td>10</td>
<td>5</td>
<td>16</td>
<td>0.80</td>
<td>1.12</td>
<td>0.71</td>
</tr>
<tr>
<td>11</td>
<td>7</td>
<td>17</td>
<td>0.62</td>
<td>1.00</td>
<td>0.62</td>
</tr>
<tr>
<td>12</td>
<td>6</td>
<td>12</td>
<td>1.98</td>
<td>0.86</td>
<td>2.29</td>
</tr>
<tr>
<td>13</td>
<td>6</td>
<td>16</td>
<td>0.67</td>
<td>0.93</td>
<td>0.72</td>
</tr>
<tr>
<td>14</td>
<td>6</td>
<td>13</td>
<td>1.49</td>
<td>0.95</td>
<td>1.57</td>
</tr>
<tr>
<td>15</td>
<td>5</td>
<td>12</td>
<td>1.21</td>
<td>1.17</td>
<td>1.03</td>
</tr>
<tr>
<td>16</td>
<td>6</td>
<td>13</td>
<td>1.58</td>
<td>0.95</td>
<td>1.66</td>
</tr>
<tr>
<td>17</td>
<td>5</td>
<td>15</td>
<td>0.82</td>
<td>0.98</td>
<td>0.84</td>
</tr>
<tr>
<td>18</td>
<td>5</td>
<td>13</td>
<td>0.89</td>
<td>0.87</td>
<td>1.01</td>
</tr>
<tr>
<td>19</td>
<td>5</td>
<td>12</td>
<td>1.25</td>
<td>1.11</td>
<td>1.13</td>
</tr>
<tr>
<td>20</td>
<td>4</td>
<td>13</td>
<td>0.99</td>
<td>1.02</td>
<td>0.96</td>
</tr>
</tbody>
</table>

To assess when the PXEM algorithm outperformed the EM algorithm, we look at $\hat{\sigma}_u^2/\hat{\sigma}^2$ in Table 8.7. The smaller $\hat{\sigma}_u^2/\hat{\sigma}^2$, the greater the difference in the number of iterations to convergence between the EM and PXEM algorithms. When $\hat{\sigma}_u^2/\hat{\sigma}^2$ was close to or greater than 1, the EM and PXEM algorithms converged in a similar number of iterations.

The AI algorithm failed for data set 8. Looking at Table 8.7, the REML estimate for $\sigma_u^2$ was small for this data set compared to the other data sets and the ratio $\hat{\sigma}_u^2/\hat{\sigma}^2$ was also small, suggesting that the AI algorithm performs poorly when $\sigma_u^2$ is small and/or when $\hat{\sigma}_u^2/\hat{\sigma}^2$ is small.

Note that the AI algorithm failed on the 2nd iteration for data set 8. Recall from Section 7.2 that we chose to define the AI algorithm as failing when the AI matrix is non-positive definite, and hence the parameter space is defined as all values of the variance parameters that result in a positive definite matrix. For data set 8, the AI matrix calculated on the 2nd iteration was non-positive definite. This was due to the AI update for $\sigma_u^2$ in the first iteration being negative. Note that $\mathbf{H}$ was not positive definite on the first iteration. The AI matrix using the updates from the first iteration was not calculated until the second iteration and so the algorithm was not terminated until the second iteration.
8.4. SIMULATED INCOMPLETE BLOCK DESIGN DATA

We now consider and compare the convergence rates of each of the algorithms for each of the data sets. Figure 8.6 shows the iteration sequence of $r_e^{(m)}$ for the AI, EM and PXEM algorithms, based on starting values of $\sigma^2_u(0) = 1$ and $\sigma^2(0) = 1$. We expect $r_e^{(m)}$ to approach the theoretical convergence rate as $m$ increases, stabilising on the theoretical convergence rate in the final iterations. Note that for data set 8, the theoretical convergence rate for AI is plotted but the iteration sequence of $r_e^{(m)}$ is not plotted since the AI algorithm failed for this data set.

For the data sets for which the AI algorithm converged, we see that $r_e^{(m)}$ approached the theoretical convergence rate over the first 3 to 5 iterations and then stabilised at the theoretical convergence rate in the last few iterations. Similar behaviour was seen for the PXEM algorithm, with $r_e^{(m)}$ rapidly approaching the theoretical convergence rate over the first 6 or so iterations, and then slowly stabilising over the last 6 or so iterations.

Some of the iteration sequences of $r_e^{(m)}$ for the EM algorithm in Figure 8.6 did not stabilise at the theoretical convergence rate, namely data sets 1, 3, 6, 7, 9, 12, 14, 15, 16, 18, 19, 20. A possible interpretation is that the algorithm did not converge. However the convergence criterion was met for these data sets, and the estimates of the variance parameters from the EM algorithm were equal to the estimates from the AI and PXEM algorithms to 7 decimal places. When the convergence criterion was tightened from $10^{-8}$ to $10^{-10}$ or $10^{-12}$ the iteration sequences get closer to stabilising to the theoretical convergence rate, and when it was tightened further to $10^{-15}$, they reached the theoretical rates but then diverged due to numerical errors. The reason for $r_e^{(m)}$ not stabilising to the theoretical convergence rate for these data sets here is something that needs to be further researched. However it highlights the importance of looking at plots of the iteration sequence of $r_e^{(m)}$ to make sure that it has stabilised, rather than just using the final value of $r_e^{(m)}$ as the measure of the empirical convergence rate.

Note that for most of the data sets $r_e^{(m)}$ for the EM and PXEM algorithms was less than or equal to that of the AI algorithm over the first few iterations, just as the AI estimates were not as close to the REML solution as the EM and PXEM estimates in the first few iterations in Figures 8.4 and 8.5. However, once the AI estimates were in the neighbourhood of the REML solution, the AI algorithm rapidly converged. These results support the use of a hybrid scheme that uses an EM type algorithm initially and then switches to AI iterations once the variance parameter estimates are within the neighbourhood of the REML solution.

Table 8.8 presents the theoretical and empirical convergence rates for the AI, EM and PXEM algorithms for the 20 data sets, rounded to two decimal places. The empirical convergence rate reported in Table 8.8 is the value of $r_e^{(m)}$ at convergence, even for the cases where the iteration sequence of $r_e^{(m)}$ did not stabilise.

The empirical convergence rates were equal to the theoretical convergence rates (rounded to two decimal places) for all data sets for the AI and PXEM algorithms, as we expected since the iteration sequence of $r_e^{(m)}$ stabilised to the theoretical convergence rate for both algorithms in Figure 8.6. The empirical convergence rate (as determined by the value of $r_e^{(m)}$ at convergence) for the EM algorithm was not equal to the theoretical convergence rate for data sets 1, 3, 7, 12, 15, 18 and 19. From the plots in 8.6, we also expected this
Figure 8.6: Theoretical convergence rate and the iteration sequence of $r^{(m)}_e$ for the AI, EM and PXEM algorithm for the 20 incomplete block design data sets with 9 replicates using $\sigma^2_u(0) = 1$ and $\sigma^2(0) = 1$ as starting values. For each algorithm, the dots represent the value of $r^{(m)}_e$ at each iteration and the horizontal line represents the theoretical convergence rate.
### Table 8.8: Theoretical and empirical convergence rates for the AI, EM and PXEM algorithms for the 20 incomplete block design data sets with 9 replicates.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Algorithm</th>
<th>REML estimates</th>
<th>Theoretical</th>
<th>Empirical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$\hat{\sigma}^2_u$</td>
<td>$\hat{\sigma}^2$</td>
<td>$\hat{\sigma}^2_u/\hat{\sigma}^2$</td>
</tr>
<tr>
<td>1</td>
<td>AI</td>
<td>1.41</td>
<td>1.13</td>
<td>1.25</td>
</tr>
<tr>
<td>2</td>
<td>EM</td>
<td>0.56</td>
<td>0.79</td>
<td>0.71</td>
</tr>
<tr>
<td>3</td>
<td>PXEM</td>
<td>1.73</td>
<td>0.88</td>
<td>1.96</td>
</tr>
<tr>
<td>4</td>
<td>EM</td>
<td>0.54</td>
<td>0.87</td>
<td>0.62</td>
</tr>
<tr>
<td>5</td>
<td>PXEM</td>
<td>0.48</td>
<td>1.06</td>
<td>0.45</td>
</tr>
<tr>
<td>6</td>
<td>EM</td>
<td>1.07</td>
<td>0.86</td>
<td>1.25</td>
</tr>
<tr>
<td>7</td>
<td>PXEM</td>
<td>1.16</td>
<td>0.66</td>
<td>1.74</td>
</tr>
<tr>
<td>8</td>
<td>EM</td>
<td>0.27</td>
<td>0.91</td>
<td>0.30</td>
</tr>
<tr>
<td>9</td>
<td>PXEM</td>
<td>1.26</td>
<td>0.98</td>
<td>1.28</td>
</tr>
<tr>
<td>10</td>
<td>EM</td>
<td>0.80</td>
<td>1.12</td>
<td>0.71</td>
</tr>
<tr>
<td>11</td>
<td>PXEM</td>
<td>0.62</td>
<td>1.00</td>
<td>0.62</td>
</tr>
<tr>
<td>12</td>
<td>EM</td>
<td>1.98</td>
<td>0.86</td>
<td>2.29</td>
</tr>
<tr>
<td>13</td>
<td>PXEM</td>
<td>0.67</td>
<td>0.93</td>
<td>0.72</td>
</tr>
<tr>
<td>14</td>
<td>EM</td>
<td>1.49</td>
<td>0.95</td>
<td>1.57</td>
</tr>
<tr>
<td>15</td>
<td>PXEM</td>
<td>1.21</td>
<td>1.17</td>
<td>1.03</td>
</tr>
<tr>
<td>16</td>
<td>EM</td>
<td>1.58</td>
<td>0.95</td>
<td>1.66</td>
</tr>
<tr>
<td>17</td>
<td>PXEM</td>
<td>0.82</td>
<td>0.98</td>
<td>0.84</td>
</tr>
<tr>
<td>18</td>
<td>EM</td>
<td>0.89</td>
<td>0.87</td>
<td>1.01</td>
</tr>
<tr>
<td>19</td>
<td>PXEM</td>
<td>1.25</td>
<td>1.11</td>
<td>1.13</td>
</tr>
<tr>
<td>20</td>
<td>EM</td>
<td>0.99</td>
<td>1.02</td>
<td>0.96</td>
</tr>
</tbody>
</table>

As expected, the theoretical and empirical convergence rates for the AI algorithm for all data sets (except for data set 8) were approximately 0, indicating approximately quadratic convergence. Note that the theoretical convergence rate for the AI algorithm for data set 8 was 0.00, however the AI algorithm failed to converge using starting values of $\sigma^2_u = 1$ and $\sigma^2 = 1$. The AI algorithm is not guaranteed to converge and the theoretical convergence rate is the rate in the neighbourhood of the REML estimate. We will see shortly that the AI algorithm converges for this data set when better starting values are used.

The convergence rates for the EM and PXEM algorithms were larger than for the AI algorithm, indicating slower convergence. The convergence rate for the PXEM algorithm for all 20 data sets was smaller than that for the EM algorithm. The convergence rates for the EM algorithm for the different data sets ranged from 0.22 to 0.54, compared to 0.21 to 0.27 for the PXEM algorithm. Hence there was less variability in the convergence rates for the PXEM algorithm than for the EM algorithm. Note that there was also less variability in the number of iterations to convergence for the PXEM algorithm (10 to 13)
compared to the EM (12 to 27) algorithm.

Table 8.8 also contains the REML estimates, \( \hat{\sigma}_u^2 \) and \( \hat{\sigma}^2 \), and their ratio \( \frac{\hat{\sigma}_u^2}{\hat{\sigma}^2} \). Note that the convergence rate for the PXEM algorithm was noticeably faster than for the EM algorithm when \( \frac{\hat{\sigma}_u^2}{\hat{\sigma}^2} \) was less than 1 and hence \( \hat{\sigma}^2 > \hat{\sigma}_u^2 \). However, when \( \frac{\hat{\sigma}_u^2}{\hat{\sigma}^2} \) was greater than 1 and hence \( \hat{\sigma}_u^2 > \hat{\sigma}^2 \), the convergence rates of the EM and PXEM algorithms were similar.

We will now look at using hybrid schemes to analyse this data set. We will look at the EM/AI, PXEM/AI, local EM/AI, local PXEM/AI schemes with the AI, update and score criteria. To be able to use the hybrid schemes with the update or score criteria, we need to decide what values to use as the cut-offs. To use the local hybrid schemes we need to decide on the number of internal local iterations to do.

Firstly, we will consider the number of internal local iterations needed for the local hybrid schemes. We have chosen to do this using the AI criterion as it does not require us to select a cut-off as needed for the update and score criteria. The data were analysed using from 1 to 10 internal local iterations for both the local EM/AI and local PXEM/AI schemes with the AI criterion. The results are presented in Table 8.9 and plotted in Figure 8.7. Recall that with the AI criterion, we start with an AI iteration and if the AI algorithm fails, we restart with an EM type update. For data set 8, the AI algorithm fails on the second iteration and so the iteration process was restarted with a local iteration.

Table 8.9: The number of iterations to convergence for the analysis of data set 8 of the incomplete block design data sets with 9 replicates, using the local EM/AI and local PXEM/AI schemes with the AI criterion with from 1 to 10 internal local iterations, using starting values of \( \sigma_u^2(0) = 1 \) and \( \sigma^2(0) = 1 \).

<table>
<thead>
<tr>
<th>Number of internal local iterations</th>
<th>local EM/AI</th>
<th>local PXEM/AI</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8 (1/7)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td>2</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td>3</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td>4</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td>5</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td>6</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td>7</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td>8</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td>9</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td>10</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
</tbody>
</table>

If we do one internal local EM iteration, the first iteration after restarting is a local EM iteration, followed by 7 AI iterations, resulting in a total of 8 iterations. This is represented as "8 (1/7)" in Table 8.9. All of the iteration sequences for local EM/AI and local PXEM/AI only required one iteration of local scheme on the first iteration. The updates from these various local schemes were close enough to the REML estimate to ensure convergence with the AI algorithm in the following iterations.
Figure 8.7: The iteration sequences for $\sigma^2_u$ and $\sigma^2$ over the first 4 iterations using the local EM/AI and local PXEM/AI schemes with the AI criterion with varying numbers of internal local iterations for the analysis of data set 8 (of the incomplete block design data sets with 9 replicates), using starting values of $\sigma^2_u(0) = 1$ and $\sigma^2(0) = 1$. For each sequence, the points are labelled with the number corresponding to the number of internal local iterations used.

Figure 8.7 shows the iteration sequences for $\sigma^2_u$ and $\sigma^2$ over the first 4 iterations for the local EM/AI and local PXEM/AI schemes with the number of internal local iterations varying from 1 to 10. Note that for each sequence, the first iteration was the local iteration, and the remaining iterations were AI iterations. The plot of the iteration sequences for $\sigma^2_u$ for the local EM/AI schemes indicates that a local EM iteration with more than 1 internal local iteration, took $\sigma^2_u$ closer to the REML estimate than a local EM iteration with only 1 local iteration. Whereas, the plot of the iteration sequences for $\sigma^2_u$ for the local PXEM/AI algorithm indicates that a local PXEM iteration with a single internal local iteration took $\sigma^2_u$ closer to the REML estimate than a local PXEM iteration with more than 1 internal local iteration. The number of internal local iterations had very little effect on the iteration sequence for $\sigma^2$ for both the local EM/AI and the local PXEM/AI algorithms.

It is important to note that although the plot of the iteration sequence for $\sigma^2_u$ for the local EM/AI algorithm indicates that a local EM iteration with 3 internal local iterations, took $\sigma^2_u$ closer to the REML estimate than a local EM iteration with 2 internal local iterations, the algorithms converged in the same number of iterations. The local EM iteration is counted as one iteration, however a local EM iteration with 2 internal local iterations will take less time than a local EM iteration with 3 internal local iterations. Seeing as both
algorithms converged in the same number of "external" iterations, it is preferable to use 2 internal local iterations for the local EM/AI scheme and 1 internal local iteration for the local PXEM/AI scheme.

We will now investigate the cut-offs for $\Delta^{(m)}$ for the update criterion and for $p_{S^{(m)}}$ for the score criterion. We will use the EM/AI scheme to investigate the sequence of iteration types for the various cut-offs. Each iteration of the hybrid scheme will begin with an AI iteration and then check to see if the AI update is within the parameter space (i.e. the variance parameter estimates result in a positive definite AI matrix) and if the update or score criterion has been met. If the conditions have been met, the AI update is used. If either of these conditions are not met, then an EM iteration is performed. If an EM iteration has previously been done, the estimates from the previous EM iteration are used. The hybrid schemes with the update and score criteria are designed to detect convergence problems of the AI algorithm before they occur and so we are looking for a cut-off for the update and score criteria that detects the convergence problems with the AI criterion as soon as possible, preferably in the first iteration.

To use the hybrid schemes with the update criterion we need to decide on a cut-off for $\Delta^{(m)}$ as defined in (7.1). That is, we need to decide on how big a change in the relative difference between consecutive updates is too big and suggests that the AI update may not be heading in the right direction. If an AI iteration results in $\Delta^{(m)}$ greater than the cut-off or variance parameter estimates that are outside the parameter space (i.e. result in a non-positive definite AI matrix), an EM iteration is used. The data were analysed using the EM/AI scheme with the update criterion with the cut-off for $\Delta^{(m)}$ ranging from 0.1 to 1.5 in increments of 0.1. The results are presented in Table 8.10.

Recall that the AI algorithm converged for all data sets except for data set 8. The minimum value that invoked EM iterations for this data set only was 1.1. A cut-off of less than 1.1 invoked EM iterations for some data sets for which the AI algorithm was able to converge. In these cases, we know that the EM iterations were invoked due to $\Delta^{(m)}$ being greater than the cut-off. However, for data set 8, we need to look at the iteration sequences for the various cut-offs to see if the EM iterations were invoked due to $\Delta^{(m)}$ being greater than the cut-off or due to the AI updates being outside of the parameter space. Table 8.11 shows the sequence of the iteration types for the EM/AI scheme for data set 8 for the various cut-offs for the update criterion.

For the EM/AI scheme with the update criterion for data set 8, when the cut-off for $\Delta^{(m)}$ was between 0.1 and 0.3, the first iteration started with an AI iteration however $\Delta^{(1)}$ was greater than the cut-off indicating that the relative change in the estimates was too large and so an EM iteration was done. The second iteration then started with an AI iteration, but again $\Delta^{(2)}$ was greater than the cut-off and an EM iteration was used. The third iteration began with an AI iteration, and $\Delta^{(3)}$ was less than the cut-off and so the AI iteration was used. AI iterations were then used until convergence. When the cut-off was between 0.4 and 1.4, the relative change in the estimates from the first AI iteration was too large and so an EM iteration was used. The AI algorithm was then able to be used for the remaining iterations since $\Delta^{(m)}$ was lower than the cut-off. When the cut-off was 1.5, $\Delta^{(1)}$ was less than 1.5 and so an AI iteration was done. The second iteration started with an AI iteration however the AI updates were not within the parameter space (i.e
8.4. SIMULATED INCOMPLETE BLOCK DESIGN DATA

Table 8.10: The results of the analyses of the 20 incomplete block design data sets with 9 replicates using the EM/AI scheme with the update criterion with various cut-offs for $\Delta^{(m)}$, using starting values of $\sigma_u^2(0) = 1$ and $\sigma^2(0) = 1$. A tick indicates that at least one EM iteration was used. A cross indicates that all iterations were AI iterations.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Cut-off for $\Delta^{(m)}$</th>
<th>0.1</th>
<th>0.2</th>
<th>0.3</th>
<th>0.4</th>
<th>0.5</th>
<th>0.6</th>
<th>0.7</th>
<th>0.8</th>
<th>0.9</th>
<th>1.0</th>
<th>1.1</th>
<th>1.2</th>
<th>1.3</th>
<th>1.4</th>
<th>1.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>8</td>
<td>fail</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>9</td>
<td>5</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>10</td>
<td>5</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>11</td>
<td>7</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>12</td>
<td>6</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>13</td>
<td>6</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>14</td>
<td>6</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>15</td>
<td>5</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>16</td>
<td>6</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>17</td>
<td>5</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>18</td>
<td>5</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>19</td>
<td>5</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>20</td>
<td>4</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 8.11: The sequence of iteration types for data set 8 of the incomplete block design data sets with 9 replicates, obtained using the EM/AI scheme with the update criterion with varying cut-offs for $\Delta^{(m)}$, using starting values of $\sigma_u^2(0)$ and $\sigma^2(0)$.

<table>
<thead>
<tr>
<th>Cut-off for $\Delta^{(m)}$</th>
<th>Iteration Sequence</th>
<th>Number of Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1 - 0.3</td>
<td>2 EM, 5 AI</td>
<td>7</td>
</tr>
<tr>
<td>0.4 - 1.4</td>
<td>EM, 7 AI</td>
<td>8</td>
</tr>
<tr>
<td>1.5</td>
<td>AI, EM*, 6 AI</td>
<td>8</td>
</tr>
</tbody>
</table>

the AI matrix was not positive definite) and so an EM iteration was done in its place. Therefore a cut-off of 1.5 was not small enough to detect convergence problems before they occurred and an EM iteration was not invoked until the AI updates were outside the
parameter space. This is identified through the use of EM in Table 8.11.

It is preferable to detect convergence problems in the first iteration and start with an EM iteration. The estimates from the EM iteration then act as good starting values for the AI algorithm which converges rapidly in the neighbourhood of the true values. Hence we want a cut-off that is small enough to detect the convergence problems in the first iteration, but that is large enough to allow the AI algorithm to allow us to swap back to the AI algorithm as soon as possible. Therefore the results in Tables 8.10 and 8.11 suggest that a cut-off for $\Delta^{(m)}$ between 1.1 and 1.5 inclusive is needed.

To use the hybrid schemes with the score criterion we need to decide on the cut-off for $p_{S(m)}$, the p-value for the proximity measure defined in (7.2). That is, we need to decide on when $p_{S(m)}$ is small enough to suggest that update is too far from the REML solution. If an AI iteration results in $p_{S(m)}$ less than the cut-off or variance parameter estimates that are outside the parameter space (i.e. result in a non-positive definite AI matrix), we invoke an EM iteration. The data were analysed using the EM/AI scheme with the score criteria with a cut-off ranging from 0.0001 to 0.5. The results are presented in Table 8.12.

Table 8.12: The results of the analyses of the 20 incomplete block design data sets with 9 replicates using the EM/AI scheme with the score criterion with various cut-offs for $p_{S(m)}$, using starting values of $\sigma^2(0)$ and $\sigma^2(0)$. A tick indicates that at least one EM iteration was used. A cross indicates that all iterations were AI iterations.
Table 8.13 shows the sequence of the iterations for the EM/AI scheme with the score criterion for data set 8 for the various cut-offs for $p_{S(m)}$. A cut-off of less than or equal to 0.01 failed to invoke an EM iteration due to $p_{S(m)}$ being less than the cut-off. An EM iteration was invoked on the second iteration due to the AI updates being outside of the parameter space (i.e. the AI matrix as non-positive definite). Whereas with a cut-off of at least 0.05, $p_{S(1)}$ based on the first AI iteration was less than the cut-off and an EM iteration was invoked.

Table 8.13: The sequence of iteration types for data set 8 (of the incomplete block design data sets with 9 replicates) obtained using the EM/AI scheme with the score criterion with varying cut-offs for $p_{S(m)}$, using starting values of $\sigma^2_u(0)$ and $\sigma^2(0)$. "EM" denotes an EM iteration that was invoked due to $p_{S(m)} < \text{cut-off}$, and "EM*" denotes an EM iteration that was invoked due to the AI updates being outside of the parameter space (i.e the AI matrix was non-positive definite).

<table>
<thead>
<tr>
<th>Cut-off for $p_{S(m)}$</th>
<th>Iteration Sequence</th>
<th>Number of Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0001 - 0.01</td>
<td>AI, EM*, 6 AI</td>
<td>8</td>
</tr>
<tr>
<td>0.05 - 0.5</td>
<td>EM, 7 AI</td>
<td>8</td>
</tr>
</tbody>
</table>

Note that when a cut-off of 0.05 was used for $p_{S(m)}$, EM iterations were also invoked for data sets 5 and 7, for which the AI algorithm was able to converge. There is no harm in doing EM iterations initially for these data sets, but it is interesting to note that we are able to find a cut-off for $\Delta^{(m)}$ for the update criterion that successfully detected the convergence problems of the AI algorithm for data set 8 but didn’t unnecessarily invoke EM iterations for the data sets where the AI algorithm successfully converged.

Now that we have decided on the number of internal local iterations to use for the local schemes (2 for local EM/AI and 1 for local PXEM/AI) and the cut-offs for the update and score criteria for the hybrid schemes (use an EM type iteration if $\Delta^{(m)} > 1.1$ for the update criterion and if $p_{S(m)} < 0.05$ for the score criterion), we will compare the results of the analyses of data set 8 for all of the hybrid schemes. The results are presented in Table 8.14.

Recall that the hybrid schemes with the AI criterion only use an EM type update when an AI update produces estimates of the variance parameters that are outside the parameter space (i.e. the variance parameter estimates result in a non-positive definite AI matrix). The first time an AI iteration fails, an EM iteration is done using the initial starting values. If a subsequent AI iteration fails, an EM iteration is done using the estimates from the previous EM iteration. Hence the results of the hybrid schemes with the AI criterion provide us with the minimum number of EM iterations that are needed to obtain estimates that are close enough to the REML solution to allow the AI algorithm to converge.

The hybrid schemes with the update and score criteria are designed to try and detect convergence problems with AI before it fails by assessing how close the iteration process is to convergence. They do not use the initial starting values for the first EM iteration,
Table 8.14: The number of iterations for the analysis of data set 8 (of the incomplete block design data sets with 9 replicates) using the various hybrid schemes with the different criteria, using starting values of $\sigma^2_u(0)$ and $\sigma^2(0)$. 2 internal local iterations were used for the local EM/AI scheme, 1 internal local iteration was used for the local PXEM/AI scheme, a cut-off of 1.1 was used for $\Delta^{(m)}$ for the update criterion and a cut-off of 0.05 was used for $p_{S^{(m)}}$ for the score criterion.

<table>
<thead>
<tr>
<th>Hybrid scheme</th>
<th>AI criterion</th>
<th>Update criterion</th>
<th>Score criterion</th>
</tr>
</thead>
<tbody>
<tr>
<td>EM/AI</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
</tr>
<tr>
<td>PXEM/AI</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td>local EM/AI</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td>local PXEM/AI</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
</tbody>
</table>

unless it is invoked in the first iteration. This was done to allow us to determine if the update or score criteria were able to detect convergence problems before they occurred. The results of the hybrid schemes with the AI criterion have been included for comparison with the schemes with the update and score criteria. Ideally the final sequence of iteration types for the schemes with the update and score criteria should match that of the schemes with the AI criterion.

Using the hybrid schemes with the AI criterion, AI failed on the 2nd iteration and the iteration sequence was restarted with an EM type iteration. The estimates from this EM type iteration were close enough to the REML solution to then allow AI iterations to be used until convergence for all of the hybrid schemes. The hybrid schemes with the update and score criteria were able to detect the convergence problems with the AI algorithm on the first iteration and use an EM type iteration for the first iteration. The results are equivalent for the update and score criteria. However, as previously discussed, note that the update criterion only invoked EM type iterations for the data set for which the AI algorithm failed to converge, whereas the score criterion invoked EM type iterations for some data sets for which the AI algorithm successfully converged.

Finally, Table 8.15 provides a summary of the number of iterations to convergence for data set 8 for the AI, EM and PXEM algorithms, and the update criterion hybrid schemes (with a cut-off of 1.1 for $\Delta^{(m)}$). The AI algorithm failed for data set 8, the EM algorithm converged in 27 iterations and the PXEM algorithm converged in 13 iterations. The hybrid schemes converged in fewer iterations than the EM and PXEM algorithms, with the PXEM/AI, local EM/AI and local PXEM/AI schemes out performing the EM/AI scheme, but only by 2 iterations.

The initial EM type iterations used in the hybrid schemes are merely a tool to move close enough to the REML solution to enable successful AI iterations. An obvious alternative is to ensure that initial values are sufficiently close to the solution at allow that AI algorithm to converge. The results of the analyses of the 20 data sets with the design shown in Table 8.5 so far have been based on starting values of $\sigma^2_u(0) = 1$ and $\sigma^2(0) = 1$. We will now look at the results of the analyses of these data sets using the AI, EM and
Table 8.15: The number of iterations for the analysis of data set 8 (of the incomplete block design data sets with 9 replicates) using the AI, EM, PXEM algorithms as well as the EM/AI, PXEM/AI, local EM/AI and local PXEM/AI schemes with the update criterion with a cut-off of 1.1 for $\Delta^{(m)}$.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Number of Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>AI</td>
<td>fail</td>
</tr>
<tr>
<td>EM</td>
<td>27</td>
</tr>
<tr>
<td>PXEM</td>
<td>13</td>
</tr>
<tr>
<td>EM/AI</td>
<td>8 (1/7)</td>
</tr>
<tr>
<td>PXEM/AI</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td>local EM/AI</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td>local PXEM/AI</td>
<td>6 (1/5)</td>
</tr>
</tbody>
</table>

PXEM algorithms with better starting values. Henderson’s Method III (HMIII) was used to calculate estimates of $\sigma_u^2$ and $\sigma^2$ that were used as starting values. Table 8.16 contains the results of the original analyses and the analyses using HMIII start values, as well as the HMIII start values and the REML estimates of $\sigma_u^2$ and $\sigma^2$.

With the starting values calculated using Henderson’s Method III, the AI algorithm converged for all 20 of the data sets. As expected, the EM and PXEM algorithms converged for all data sets. Using the informed starting values decreased the number of iterations to convergence for the AI, EM and PXEM algorithms for all data sets, highlighting the fact that good starting values are important. Again we see here that the AI algorithm converged in half the number of iterations that the PXEM algorithm converged in, and the EM algorithm converged in more iterations than the PXEM algorithm. Note that the AI algorithm converged for data set 8 when better starting values were used. And it converged in 3 iterations which is better than the result obtained with any of the hybrid schemes.
Table 8.16: The number of iterations to convergence for the AI, EM and PXEM algorithms for the 20 incomplete block design data sets with 9 replicates using the uninformed starting values $\sigma^2_u (0)$ and $\sigma^2 (0)$ and the starting values calculated via Henderson’s Method III.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Number of Iterations Using Uninformed Starting Values</th>
<th>Number of Iterations Using Informed Starting Values</th>
<th>HMIII Starting Values</th>
<th>REML estimates</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AI  EM  PXEM</td>
<td>AI  EM  PXEM</td>
<td>$\sigma^2_u (0)$</td>
<td>$\sigma^2 (0)$</td>
</tr>
<tr>
<td>1</td>
<td>6     13   12</td>
<td>4    10   8</td>
<td>1.42</td>
<td>1.13</td>
</tr>
<tr>
<td>2</td>
<td>8     16   13</td>
<td>4    13   9</td>
<td>0.56</td>
<td>0.79</td>
</tr>
<tr>
<td>3</td>
<td>6     13   12</td>
<td>4    9    8</td>
<td>1.74</td>
<td>0.88</td>
</tr>
<tr>
<td>4</td>
<td>7     17   12</td>
<td>3    11   7</td>
<td>0.54</td>
<td>0.87</td>
</tr>
<tr>
<td>5</td>
<td>8     21   13</td>
<td>3    14   8</td>
<td>0.48</td>
<td>1.06</td>
</tr>
<tr>
<td>6</td>
<td>5     13   12</td>
<td>4    11   8</td>
<td>1.06</td>
<td>0.86</td>
</tr>
<tr>
<td>7</td>
<td>6     13   13</td>
<td>3    8    6</td>
<td>1.16</td>
<td>0.66</td>
</tr>
<tr>
<td>8</td>
<td>fail   27   13</td>
<td>3    18   9</td>
<td>0.27</td>
<td>0.91</td>
</tr>
<tr>
<td>9</td>
<td>5     13   11</td>
<td>4    10   8</td>
<td>1.27</td>
<td>0.98</td>
</tr>
<tr>
<td>10</td>
<td>5     16   12</td>
<td>4    13   9</td>
<td>0.80</td>
<td>1.12</td>
</tr>
<tr>
<td>11</td>
<td>7     17   12</td>
<td>4    14   9</td>
<td>0.63</td>
<td>1.00</td>
</tr>
<tr>
<td>12</td>
<td>6     12   12</td>
<td>3    8    7</td>
<td>1.98</td>
<td>0.86</td>
</tr>
<tr>
<td>13</td>
<td>6     16   11</td>
<td>3    12   8</td>
<td>0.67</td>
<td>0.93</td>
</tr>
<tr>
<td>14</td>
<td>6     13   11</td>
<td>4    10   8</td>
<td>1.47</td>
<td>0.95</td>
</tr>
<tr>
<td>15</td>
<td>5     12   12</td>
<td>3    10   8</td>
<td>1.22</td>
<td>1.17</td>
</tr>
<tr>
<td>16</td>
<td>6     13   11</td>
<td>4    10   8</td>
<td>1.59</td>
<td>0.95</td>
</tr>
<tr>
<td>17</td>
<td>5     15   10</td>
<td>3    11   8</td>
<td>0.83</td>
<td>0.98</td>
</tr>
<tr>
<td>18</td>
<td>5     13   12</td>
<td>3    11   8</td>
<td>0.88</td>
<td>0.87</td>
</tr>
<tr>
<td>19</td>
<td>5     12   12</td>
<td>4    11   9</td>
<td>1.27</td>
<td>1.11</td>
</tr>
<tr>
<td>20</td>
<td>4     13   11</td>
<td>4    11   8</td>
<td>0.99</td>
<td>1.02</td>
</tr>
</tbody>
</table>
8.4. SIMULATED INCOMPLETE BLOCK DESIGN DATA

8.4.2 Incomplete Block Design with 3 replicates

We will now look at the results of the analyses of the 20 data sets with the design shown in Table 8.6, that is the incomplete block design with 3 replicates. The first set of analyses were performed using $\sigma_u^2(0) = 1$ and $\sigma^2(0) = 1$ as starting values. The number of iterations to convergence for the AI, EM and PXEM algorithms, as well as the REML estimates of $\sigma_u^2$ and $\sigma^2$, for each of the 20 data sets are given in Table 8.17. The iteration sequences for $\sigma_u^2$ and $\sigma^2$ for each of the algorithms for each of the data sets are given in Figures 8.8 and 8.9 respectively.

Table 8.17: The number of iterations to convergence for the AI, EM and PXEM algorithms for the 20 incomplete block design data sets with 3 replicates using $\sigma_u^2(0) = 1$ and $\sigma^2(0) = 1$ as starting values.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Algorithm</th>
<th>REML estimates</th>
<th>$\hat{\sigma}_u^2$</th>
<th>$\hat{\sigma}^2$</th>
<th>$\hat{\sigma}_u^2/\hat{\sigma}^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>fail</td>
<td>45 38</td>
<td>0.31</td>
<td>0.49</td>
<td>0.63</td>
</tr>
<tr>
<td>2</td>
<td>11</td>
<td>55 48</td>
<td>1.36</td>
<td>0.94</td>
<td>1.45</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>103 45</td>
<td>0.35</td>
<td>1.12</td>
<td>0.32</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>38 32</td>
<td>1.41</td>
<td>1.03</td>
<td>1.36</td>
</tr>
<tr>
<td>5</td>
<td>11</td>
<td>248 82</td>
<td>0.28</td>
<td>1.28</td>
<td>0.21</td>
</tr>
<tr>
<td>6</td>
<td>9</td>
<td>95 56</td>
<td>0.66</td>
<td>1.38</td>
<td>0.48</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>40 39</td>
<td>1.16</td>
<td>0.63</td>
<td>1.85</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>60 41</td>
<td>0.75</td>
<td>1.27</td>
<td>0.59</td>
</tr>
<tr>
<td>9</td>
<td>fail</td>
<td>133 50</td>
<td>0.25</td>
<td>0.91</td>
<td>0.27</td>
</tr>
<tr>
<td>10</td>
<td>8</td>
<td>86 52</td>
<td>0.67</td>
<td>1.32</td>
<td>0.51</td>
</tr>
<tr>
<td>11</td>
<td>10</td>
<td>105 59</td>
<td>0.76</td>
<td>1.72</td>
<td>0.44</td>
</tr>
<tr>
<td>12</td>
<td>6</td>
<td>43 40</td>
<td>1.79</td>
<td>1.61</td>
<td>1.11</td>
</tr>
<tr>
<td>13</td>
<td>fail</td>
<td>5491 301</td>
<td>0.03</td>
<td>1.49</td>
<td>0.02</td>
</tr>
<tr>
<td>14</td>
<td>7</td>
<td>56 40</td>
<td>0.82</td>
<td>1.30</td>
<td>0.63</td>
</tr>
<tr>
<td>15</td>
<td>10</td>
<td>40 39</td>
<td>2.53</td>
<td>0.53</td>
<td>5.00</td>
</tr>
<tr>
<td>16</td>
<td>7</td>
<td>38 38</td>
<td>2.32</td>
<td>0.60</td>
<td>3.84</td>
</tr>
<tr>
<td>17</td>
<td>fail</td>
<td>71 42</td>
<td>0.39</td>
<td>0.75</td>
<td>0.52</td>
</tr>
<tr>
<td>18</td>
<td>7</td>
<td>39 35</td>
<td>0.99</td>
<td>0.90</td>
<td>1.09</td>
</tr>
<tr>
<td>19</td>
<td>11</td>
<td>99 61</td>
<td>0.79</td>
<td>1.51</td>
<td>0.52</td>
</tr>
<tr>
<td>20</td>
<td>8</td>
<td>35 34</td>
<td>2.82</td>
<td>0.89</td>
<td>3.16</td>
</tr>
</tbody>
</table>

The AI algorithm converged for 16 of the 20 data sets. As expected, the EM and PXEM algorithms converged for all data sets. The number of iterations to convergence for the PXEM algorithm were fewer than the number of iterations to convergence for the EM algorithm for all but one data set (data set 16). For the data sets where AI converged, it did so in noticeably fewer iterations than the EM and PXEM algorithms.

The AI algorithm failed for data sets 1, 9, 13 and 17. In each case, the AI algorithm failed on the 2nd iteration due to the AI matrix being non-positive definite. Note that in each case, $\sigma_u^2(1)$ in the first iteration was negative, and $\mathbf{H}^{(1)}$ was not positive definite. The AI matrix using the updates from the first iteration was not calculated until the
Figure 8.8: Iteration sequence for $\sigma^2_u$ for the AI, EM and PXEM algorithms for the 20 incomplete block design data sets with 3 replicates using $\sigma^2_u(0)$ and $\sigma^2(0)$ as starting values.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Number of Iterations</th>
<th>$\sigma^2_u$ Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>11</td>
<td>0.30, 0.60, 0.90</td>
</tr>
<tr>
<td>2</td>
<td>11</td>
<td>0.50, 0.80, 1.10</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>-0.10, 0.20, 0.50</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>0.30, 0.50, 0.70</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
<td>0.40, 0.60, 0.80</td>
</tr>
<tr>
<td>6</td>
<td>11</td>
<td>0.30, 0.60, 0.90</td>
</tr>
<tr>
<td>7</td>
<td>11</td>
<td>0.50, 0.80, 1.10</td>
</tr>
<tr>
<td>8</td>
<td>11</td>
<td>-0.10, 0.20, 0.50</td>
</tr>
<tr>
<td>9</td>
<td>11</td>
<td>0.30, 0.50, 0.70</td>
</tr>
<tr>
<td>10</td>
<td>8</td>
<td>0.40, 0.60, 0.80</td>
</tr>
<tr>
<td>11</td>
<td>10</td>
<td>0.30, 0.60, 0.90</td>
</tr>
<tr>
<td>12</td>
<td>3</td>
<td>0.80, 0.90, 1.00</td>
</tr>
<tr>
<td>13</td>
<td>11</td>
<td>0.30, 0.50, 0.70</td>
</tr>
<tr>
<td>14</td>
<td>11</td>
<td>0.50, 0.80, 1.10</td>
</tr>
<tr>
<td>15</td>
<td>11</td>
<td>-0.10, 0.20, 0.50</td>
</tr>
<tr>
<td>16</td>
<td>7</td>
<td>0.30, 0.60, 0.90</td>
</tr>
<tr>
<td>17</td>
<td>11</td>
<td>0.30, 0.60, 0.90</td>
</tr>
<tr>
<td>18</td>
<td>11</td>
<td>0.50, 0.80, 1.10</td>
</tr>
<tr>
<td>19</td>
<td>11</td>
<td>-0.10, 0.20, 0.50</td>
</tr>
<tr>
<td>20</td>
<td>8</td>
<td>0.40, 0.60, 0.80</td>
</tr>
</tbody>
</table>
Figure 8.9: Iteration sequence for $\sigma^2$ for the AI, EM and PXEM algorithms for the 20 incomplete block design data sets with 3 replicates using $\sigma^2_u(0)$ and $\sigma^2(0)$ as starting values.
second iteration and so the algorithm was not terminated until the second iteration. Note that this problem could have been detected earlier by terminating the AI algorithm if the update for either $\sigma^2_u$ or $\sigma^2$ were negative. However, looking at the iteration sequence of $\sigma^2_u$ for data set 3 in Figure 8.8 we see that $\sigma^2_u^{(1)}$ was negative, yet the AI algorithm still converged. If we had terminated the AI algorithm if the update for either of the variance parameters was negative, the AI algorithm would have been deemed to have failed for this data set when in fact it converged. In this case, a step in the wrong direction in the first iteration provided a better position, than the original starting values, for subsequent iterations. However, often the AI algorithm is not able to recover from steps away from the maximum of the REML log-likelihood.

Looking at Table 8.17, the AI algorithm failed when $\hat{\sigma}^2_u$ and/or $\hat{\sigma}^2$ were small. It appears that solely looking at the ratio $\hat{\sigma}^2_u/\hat{\sigma}^2$ does not explain the pattern for when the AI algorithm does and does not work. The data sets for which the AI algorithm failed to converge had a low value of $\hat{\sigma}^2_u/\hat{\sigma}^2$ but there were other data sets that also had low values of $\hat{\sigma}^2_u/\hat{\sigma}^2$ for which the AI algorithm converged. Data set 1 had small $\hat{\sigma}^2_u$ and $\hat{\sigma}^2$, data set 9 and 17 had small $\hat{\sigma}^2_u$ and reasonably small $\hat{\sigma}^2$, whereas data set 13 had very small $\hat{\sigma}^2_u$. It seems that if $\hat{\sigma}^2_u < 0.5$ but $\hat{\sigma}^2 > 1$ then AI converged unless $\hat{\sigma}^2_u$ was very small.

Recall that the AI algorithm only failed for one of the 20 data sets with 9 replicates of each treatment. For the 20 data sets with 3 replicates of each treatment, the AI algorithm failed for 4 of the data sets. For the data sets with 3 replicates for which AI converged, it did so in between 7 and 11 iterations, compared to between 4 and 8 iterations for the data sets with 9 replicates. So with fewer replicates and hence greater confounding, the AI algorithm took a few more iterations to converge.

There was a greater difference in the number of iterations to convergence for the EM and PXEM algorithms between the data sets with 9 replicates and those with 3 replicates. For the data sets with 9 replicates, the EM algorithm converged in between 13 and 27 iterations, whereas for the data sets with 3 replicates, the EM algorithm converged in between 35 and 5491 iterations. The PXEM algorithm converged in between 10 and 13 iterations for the data sets with 9 replicates, but converged in between 32 and 301 iterations for the data sets with 3 replicates. For the data sets with 9 replicates, the number of iterations to convergence for the EM and PXEM algorithms were similar, with the PXEM algorithm converging in slightly fewer iterations. As we saw for the data sets with 9 replicates, the smaller $\hat{\sigma}^2_u/\hat{\sigma}^2$, the greater the difference in the number of iterations to convergence between the EM and PXEM algorithms. However with only 3 replicates of each treatment, the difference was much greater with the PXEM algorithm converging in far fewer iterations than the EM algorithm for some of the data sets.

The plots in Figures 8.8 and 8.9 show the EM and PXEM estimates rapidly approaching the REML estimates over the first 10 or so iterations, but then taking a large number of iterations to actually converge. For the data sets with 9 replicates of each treatment we saw that the EM and PXEM updates were often closer to the REML solution than the AI updates over the early iterations. This was not the case for the data sets with 3 replicates. For the data sets where the AI algorithm converged, it did so in no more than 11 iterations. Hence for these data sets, the AI algorithm converged before or soon after the EM and PXEM estimates plateau and then slowly move towards convergence.
As expected the PXEM estimates for $\sigma_u^2$ approached the REML estimate for $\sigma_u^2$ more rapidly than the EM estimates in the early iterations, but there was little difference for $\sigma^2$. The smaller $\hat{\sigma}_u^2/\hat{\sigma}^2$, the greater the difference in the number of iterations to convergence between the EM and PXEM algorithms. When $\hat{\sigma}_u^2/\hat{\sigma}^2$ was close to or greater than 1, the EM and PXEM algorithms converged in a similar number of iterations.

We will now compare the convergence rates of each of the algorithms for each of the data sets. Figure 8.10 shows the iteration sequence of $r_e^{(m)}$ for the AI, EM and PXEM algorithms, based on starting values of $\sigma_u^2(0) = 1$ and $\sigma^2(0) = 1$. We expect $r_e^{(m)}$ to approach the theoretical convergence rate as $m$ increases, stabilising on the theoretical convergence rate in the final iterations. Table 8.18 presents the theoretical and empirical convergence rates for the AI, EM and PXEM algorithms for the 20 data sets, rounded to two decimal places. Note that for data sets for which the AI algorithm failed, the theoretical convergence rate for AI is plotted but the iteration sequence of $r_e^{(m)}$ is not plotted since the empirical convergence rate is not available.

Table 8.18: Theoretical and empirical convergence rates for the AI, EM and PXEM algorithms for the 20 incomplete block design data sets with 3 replicates.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Algorithm</th>
<th>REML estimates</th>
<th>Theoretical</th>
<th>Empirical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AI</td>
<td>EM</td>
<td>PXEM</td>
<td>$\sigma_u^2$</td>
</tr>
<tr>
<td>1</td>
<td>fail</td>
<td>45</td>
<td>38</td>
<td>0.31</td>
</tr>
<tr>
<td>2</td>
<td>11</td>
<td>55</td>
<td>48</td>
<td>1.36</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>103</td>
<td>45</td>
<td>0.35</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>38</td>
<td>32</td>
<td>1.41</td>
</tr>
<tr>
<td>5</td>
<td>11</td>
<td>248</td>
<td>82</td>
<td>0.28</td>
</tr>
<tr>
<td>6</td>
<td>9</td>
<td>95</td>
<td>56</td>
<td>0.66</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>40</td>
<td>39</td>
<td>1.16</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>60</td>
<td>41</td>
<td>0.75</td>
</tr>
<tr>
<td>9</td>
<td>fail</td>
<td>133</td>
<td>50</td>
<td>0.25</td>
</tr>
<tr>
<td>10</td>
<td>8</td>
<td>86</td>
<td>52</td>
<td>0.67</td>
</tr>
<tr>
<td>11</td>
<td>10</td>
<td>105</td>
<td>59</td>
<td>0.76</td>
</tr>
<tr>
<td>12</td>
<td>6</td>
<td>43</td>
<td>40</td>
<td>1.79</td>
</tr>
<tr>
<td>13</td>
<td>fail</td>
<td>5491</td>
<td>301</td>
<td>0.03</td>
</tr>
<tr>
<td>14</td>
<td>7</td>
<td>56</td>
<td>40</td>
<td>0.82</td>
</tr>
<tr>
<td>15</td>
<td>10</td>
<td>40</td>
<td>39</td>
<td>2.63</td>
</tr>
<tr>
<td>16</td>
<td>7</td>
<td>38</td>
<td>38</td>
<td>2.32</td>
</tr>
<tr>
<td>17</td>
<td>fail</td>
<td>71</td>
<td>42</td>
<td>0.39</td>
</tr>
<tr>
<td>18</td>
<td>7</td>
<td>39</td>
<td>35</td>
<td>0.99</td>
</tr>
<tr>
<td>19</td>
<td>11</td>
<td>99</td>
<td>61</td>
<td>0.79</td>
</tr>
<tr>
<td>20</td>
<td>8</td>
<td>35</td>
<td>34</td>
<td>2.82</td>
</tr>
</tbody>
</table>

The iteration sequence of $r_e^{(m)}$ for the AI, EM and PXEM algorithms in Figure 8.10 stabilised at the theoretical convergence rate for all data sets where the algorithms successfully converged. And hence the empirical convergence rate was equal to the theoretical convergence rate for the AI, EM and PXEM algorithms for all data sets in Table 8.18.
Figure 8.10: Theoretical convergence rate and the iteration sequence of $r_e^{(m)}$ for the AI, EM and PXEM algorithm for the 20 incomplete block design data sets with 3 replicates using $\sigma^2_u(0)$ and $\sigma^2(0)$ as starting values. For each algorithm, the dots represent the value of $r_e^{(m)}$ at each iteration and the horizontal line represents the theoretical convergence rate.
The convergence rates for the AI algorithm for the majority of the data sets were less than 0.1, with the largest being 0.15, indicating fast convergence. Note that the theoretical convergence rates for the AI algorithm for the data sets where the AI algorithm fails, range from 0.12 to 0.01, indicating that fast convergence is expected in the neighbourhood of the REML estimates. We will see shortly that the AI algorithm converged for these data sets when better starting values were used.

As expected, the convergence rates for the EM and PXEM algorithm were larger than for the AI algorithm, indicating slower convergence. The convergence rates for the PXEM algorithm for all 20 data sets were smaller than those for the EM algorithm, except for data sets 15 and 16 where they were equal. Note that for data set 13, the convergence rate for the EM algorithm was 1.00 compared to 0.96 for the PXEM algorithm, suggesting that the PXEM algorithm was marginally quicker than the EM algorithm. However when we look at the number of iterations to convergence for the two algorithms, the EM converged in 5491 iterations whereas the PXEM algorithm converged in 301 iterations indicating that its convergence was noticeably faster.

Recall for the data sets with 9 replicates of each treatment we saw that \( r_e^{(m)} \) was often smaller for the EM and PXEM algorithms than the EM algorithms over the first few iterations. For the data sets with 3 replicates, this is the case for less than half of the data sets. When the AI algorithm converged, its performance was generally superior to the EM and PXEM algorithms over the entire sequence of iterations. However in the cases where the AI algorithm did not converge, \( r_e^{(m)} \) was small for the EM and PXEM algorithms over the first few iterations, indicating that the algorithms made rapid progress in the early iterations. This was also evident in the plots of the iteration sequences for \( \sigma^2_u \) and \( \sigma^2 \) in Figures 8.8 and 8.9. These results support the use of a hybrid scheme that uses EM or PXEM iterations initially and then switches to AI iterations once the variance parameter estimates are within the neighbourhood of the REML solution.

We will now look at using the hybrid schemes for the data sets for which the AI algorithm failed, that is data sets 1, 9, 13 and 17. Note that for these 4 data sets, the AI algorithm failed on the second iteration due to the AI updates being outside of the parameter space (i.e. the AI matrix was non-positive definite). To use the local hybrid schemes we need to decide on the number of internal local iterations to do. The data were analysed using from 1 to 10 local iterations for both the local EM/AI and local PXEM/AI schemes with the AI criterion. The number of iterations to convergence for the local EM/AI and local PXEM/AI are presented in Table 8.19.

The local hybrid schemes did not work for data set 1. The iteration sequence started with an AI iteration but then failed on the second AI iteration. The process was then restarted with a local iteration, followed by an AI iteration. However, the second AI iteration failed and so another local iteration was done using the estimates from the previous local iteration. This sequence was repeated until the variance parameter estimates converged. However, the iteration sequence consisted only of local iterations with \( \sigma^2 \) held fixed at the starting value of 1 and only updating \( \sigma^2_u \). Hence they did not converge to the REML estimates, they converged to \( \sigma^2_u = 0.15 \) and \( \sigma^2 = 1 \). This could possibly be resolved by adjusting \( \sigma^2 \) after \( \sigma^2_u \) is updated at each iteration using Result A.19. This is something that requires further research.
Table 8.19: The number of iterations to convergence for data sets 9, 13 and 17 (of the incomplete block design data sets with 3 replicates) using the local EM/AI and local PXEM/AI schemes with the AI criterion with from 1 to 10 internal local iterations, using starting values of $\sigma^2_u(0)$ and $\sigma^2(0)$.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Number of internal local iterations</th>
<th>local EM/AI</th>
<th>local PXEM/AI</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>1-10</td>
<td>fail</td>
<td>fail</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>11 (1/10)</td>
<td>8 (1/7)</td>
</tr>
<tr>
<td>2</td>
<td>9 (1/8)</td>
<td>8 (1/7)</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>9 (1/8)</td>
<td>8 (1/7)</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>9 (1/8)</td>
<td>8 (1/7)</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>8 (1/7)</td>
<td>7 (1/6)</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>8 (1/7)</td>
<td>7 (1/6)</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>9 (1/8)</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td>2</td>
<td>8 (1/7)</td>
<td>7 (1/6)</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td>9 (1/8)</td>
<td>8 (1/7)</td>
</tr>
<tr>
<td>2</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>7 (1/6)</td>
<td>8 (1/7)</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td></td>
</tr>
</tbody>
</table>

For data sets 9, 13 and 17, a single local iteration was all that was needed for both the EM and PXEM local schemes to take the variance parameter estimates into the neighbourhood of the solution, and hence allow the AI algorithm to converge. The iteration sequences for $\sigma^2_u$ and $\sigma^2$ over the first 4 iterations for the various local schemes for data sets 9, 13 and 17 are presented in Figure 8.11. The plots indicate that a local EM iteration with more than 1 internal local iteration, took $\sigma^2_u$ closer to the REML estimate than a local EM iteration with only 1 local iteration. And as a result, the AI update used for the
second iteration took \( \sigma^2_u \) closer to the REML estimate. Whereas, the plots of the iteration sequences for \( \sigma^2_u \) for the local PXEM/AI algorithm indicate that a local PXEM iteration with a single internal local iteration took \( \sigma^2_u \) closer to the REML estimate than a local PXEM iteration with more than 1 internal local iteration. The number of internal local iterations had very little effect on the iteration sequences for \( \sigma^2 \) for both the local EM/AI and the local PXEM/AI algorithms. Based on these results, we have chosen to use 2 internal local iterations for the local EM/AI scheme and 1 local iteration for the local PXEM/AI scheme.

Figure 8.11: The iteration sequences for \( \sigma^2_u \) and \( \sigma^2 \) over the first 4 iterations using the local EM/AI and local PXEM/AI schemes with the AI criterion with from 1 to 10 internal local iterations for data sets 9, 13 and 17 (of the incomplete block design data sets with 3 replicates), using \( \sigma^2_u(0) \) and \( \sigma^2(0) \) as starting values. For each sequence, the points are labelled with the number corresponding to the number of internal local iterations used.

We will now look at the results of the analyses of data sets 1, 9, 13 and 17 using the hybrid schemes with the various criteria, starting with the update criterion. The data were analysed using the EM/AI scheme with the update criterion with cut-offs for \( \Delta^{(m)} \) ranging from 0.1 to 1.5 in increments of 0.1. The results are presented in Table 8.20. The minimum cut-off for \( \Delta^{(m)} \) that invoked the hybrid scheme for data sets 1, 9, 13 and 17...
only was 1.1. Note that this is the same value that was selected for the analyses of the incomplete block designs with 9 replicates.

Table 8.20: The results of the analyses of the 20 incomplete block design data sets with 3 replicates using the EM/AI scheme with the update criterion with various cut-offs for \( \Delta^{(m)} \) using starting values of \( \sigma^2(u) \) and \( \sigma^2(0) \). A tick indicates that at least one EM iteration was used. A cross indicates that all iterations were AI iterations.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>AI</th>
<th>0.1</th>
<th>0.2</th>
<th>0.3</th>
<th>0.4</th>
<th>0.5</th>
<th>0.6</th>
<th>0.7</th>
<th>0.8</th>
<th>0.9</th>
<th>1.0</th>
<th>1.1</th>
<th>1.2</th>
<th>1.3</th>
<th>1.4</th>
<th>1.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>fail</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>11</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>✔</td>
<td></td>
<td></td>
<td></td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>5</td>
<td>11</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>6</td>
<td>9</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>9</td>
<td>fail</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>10</td>
<td>8</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>11</td>
<td>10</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>12</td>
<td>6</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>13</td>
<td>fail</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>14</td>
<td>7</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>15</td>
<td>10</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>16</td>
<td>7</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>17</td>
<td>fail</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>18</td>
<td>7</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>19</td>
<td>11</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>20</td>
<td>8</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
</tbody>
</table>

To make sure that EM iterations were invoked due to \( \Delta^{(m)} \) being greater than the cut-off, rather than due at an AI iteration failing, we need to look at the sequence of iteration types for the various cut-offs. Table 8.21 shows the sequence of the iteration types for the EM/AI scheme with the update criterion for data sets 1, 9, 13 and 17 for the various cut-offs for \( \Delta^{(m)} \). Recall that "EM*" denotes an EM iteration that is invoked due to the AI update being outside of the parameter space (i.e. the AI matrix was non-positive definite), rather than due to \( \Delta^{(m)} \) being larger than the cut-off. Sequences containing an EM* indicate that the update criterion did not detect convergence problems before they occurred.

The results indicate that if the cut-off is too low, an unnecessary number of EM iterations are done. For example the iteration sequence for data set 13 with a cut-off of 0.1 consisted of 29 EM iterations before switching to AI iterations, whereas a cut-off in the range 0.8 to 1.1 resulted in a single EM iteration before switching to AI iterations. If the cut-off is too high, convergence problems are not detected and EM iterations are only invoked when the AI algorithm fails. For example, the iteration sequence for data set 13 with
Table 8.21: The sequence of iteration types for data sets 1, 9, 13 and 17 (of the incomplete block design data sets with 3 replicates) obtained using the EM/AI scheme with the update criterion with varying cut-offs for $\Delta^{(m)}$, using starting values of $\sigma^2_u(0)$ and $\sigma^2(0)$. “EM” denotes an EM iteration invoked due to $\Delta^{(m)} > \text{cut-off}$, and “EM*” denotes an EM iteration invoked due to the estimates from an AI update being outside the parameter space (i.e. the AI matrix was non-positive definite).

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Cut-off for $\Delta^{(m)}$</th>
<th>Iteration Sequence</th>
<th>Number of Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.1</td>
<td>4 EM, 9 AI</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td>0.2-0.5</td>
<td>3 EM, 9 AI</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>0.6-1.5</td>
<td>2 EM, 10 AI</td>
<td>12</td>
</tr>
<tr>
<td>9</td>
<td>0.2</td>
<td>4 EM, 7 AI</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>0.2</td>
<td>3 EM, 8 AI</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>0.3-0.7</td>
<td>2 EM, 8 AI</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>0.8-1.4</td>
<td>1 EM, 9 AI</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>1.5</td>
<td>AI, 3 EM*, 2 EM, 9 AI</td>
<td>15</td>
</tr>
<tr>
<td>13</td>
<td>0.1</td>
<td>29 EM, 5 AI</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>0.2</td>
<td>10 EM, 6 AI</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>0.3</td>
<td>5 EM, 7 AI</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>0.4</td>
<td>4 EM, 7 AI</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>0.5</td>
<td>3 EM, 7 AI</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>0.6-0.7</td>
<td>2 EM, 7 AI</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>0.8-1.1</td>
<td>EM, 8 AI</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>1.2-1.5</td>
<td>AI, 3 EM*, 2 EM, 9 AI</td>
<td>16</td>
</tr>
<tr>
<td>17</td>
<td>0.1</td>
<td>4 EM, 6 AI</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>0.2-0.6</td>
<td>2 EM, 7 AI</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>0.7-1.5</td>
<td>EM, 7 AI</td>
<td>8</td>
</tr>
</tbody>
</table>

A cut-off of 1.2 started with an AI iteration but the 2nd AI iteration failed and an EM iteration was used. The scheme then used AI iterations but failed again on the second AI iteration and so an EM iteration based on the values form the previous EM iteration was used. This pattern was repeated once more and then two more EM iterations were used due to $\Delta^{(m)}$ being greater than the cut-off before the scheme successfully switched to AI iterations and converged.

As we have previously stated, it is preferable to detect convergence problems in the first iteration and start with an EM iteration. Hence we want a cut-off for $\Delta^{(m)}$ that is small enough to detect the convergence problems before they occur, but that is large enough to allow us to switch to the AI algorithm as soon as possible. Therefore the results in Table 8.11 support our selection of 1.1 as the cutoff for $\Delta^{(m)}$ for the hybrid schemes with the update criterion.

To use the hybrid schemes with the score criterion we need to decide on the cut-off for
The data were analysed with a cut-off ranging from 0.0001 to 0.5. The results are presented in Table 8.22. The maximum cut-off for $p_{S(m)}$ that invoked EM iterations only for the data sets for which the AI algorithm failed to converge was 0.01. To assess whether the EM iterations were invoked due to $p_{S(m)}$ being less than the cut-off or due to the AI update being outside the parameter space (i.e. the AI matrix was non-positive definite), we need to look at the sequence of iteration types.

Table 8.22: The results of the analyses of the 20 incomplete block design data sets with 3 replicates using the hybrid scheme with the score criterion using various cut-offs for $p_{S(m)}$, using starting values of $\sigma^2_u(0)$ and $\sigma^2(0)$. A tick indicates that at least one EM iteration was used. A cross indicates that all iterations were AI iterations.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>AI</th>
<th>0.0001</th>
<th>0.0005</th>
<th>0.001</th>
<th>0.005</th>
<th>0.01</th>
<th>0.05</th>
<th>0.1</th>
<th>0.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>fail</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>2</td>
<td>11</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>5</td>
<td>11</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>6</td>
<td>9</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>9</td>
<td>fail</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>10</td>
<td>8</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>11</td>
<td>10</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>12</td>
<td>6</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>13</td>
<td>fail</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>14</td>
<td>7</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>15</td>
<td>10</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>16</td>
<td>7</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>17</td>
<td>fail</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>18</td>
<td>7</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>19</td>
<td>11</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>20</td>
<td>8</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 8.23 shows the sequence of the iteration types for the EM/AI scheme with the score criterion for the various cut-offs for $p_{S(m)}$ for data sets 1, 9, 13 and 17. If the AI algorithm is not going to converge, it is preferable to detect this as early as possible and start the iteration process using an EM iteration to obtain variance parameter estimates that are closer to the REML solution and hence will increase the chance of the AI algorithm converging. A cut-off of at least 0.5 was needed to invoke EM iterations before the AI algorithm failed. Note that a cut-off of 0.5 also invoked EM iterations for some of the data sets for which the AI algorithm successfully converged. There is no harm in doing a EM iterations initially for these data sets. However it is again interesting to note that we are able to find a cut-off for the update criterion that successfully detects the convergence problems of the AI algorithm but does not unnecessarily invoke EM iterations for the
Table 8.23: The sequence of iteration types for data sets 1, 9, 13 and 17 (of the incomplete block design data sets with 3 replicates) obtained using the EM/AI scheme with the score criterion with varying cut-offs for $p_{S(m)}$. "EM" denotes an EM iteration invoked due to $p_{S(m)} <$ cut-off, and "EM*" denotes an EM iteration invoked due to the estimates from an AI update being outside the parameter space (i.e. the AI matrix was non-positive definite).

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Cut-off for $p_{S(m)}$</th>
<th>Iteration Sequence</th>
<th>Number of Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.0001-0.005</td>
<td>AI, EM*, 11 AI</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td>0.01</td>
<td>EM, 12 AI</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td>0.05-0.1</td>
<td>2 EM, 10 AI</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>0.5</td>
<td>2 EM, 10 AI</td>
<td>12</td>
</tr>
<tr>
<td>9</td>
<td>0.0001-0.001</td>
<td>AI, EM*, 2 EM, 2 EM*, 9 AI</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>0.005-0.1</td>
<td>AI, EM*, 3 EM, EM*, 9 AI</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>0.5</td>
<td>2 EM, 8 AI</td>
<td>10</td>
</tr>
<tr>
<td>13</td>
<td>0.0001-0.05</td>
<td>AI, EM*, 9 EM, EM*, 9 AI</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>0.1</td>
<td>AI, EM*, 4 EM, EM*, EM, 7 AI</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>0.5</td>
<td>2 EM, 7 AI</td>
<td>9</td>
</tr>
<tr>
<td>17</td>
<td>0.0001-0.001</td>
<td>AI, EM*, 9 EM, EM*, 9 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>0.005-0.1</td>
<td>AI, EM*, 12 EM, 7 AI</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>0.5</td>
<td>EM, 7 AI</td>
<td>8</td>
</tr>
</tbody>
</table>

data sets where the AI algorithm successfully converged. Also note that we chose to use a cut-off of 0.05 for $p_{S(m)}$ for the analysis of the simulated incomplete block design data sets with 9 replicates. We could use a cut-off of 0.5 for the analysis of these data sets and as a result invoke EM iterations for a few more data sets where the AI algorithm successfully converged.

Now that we have decided on the number of internal local iterations to use for the local schemes (2 for local EM/AI and 1 for local PXEM/AI) and the cut-offs for the update and score criterion for the hybrid schemes (use an EM type iteration if $\Delta^{(m)} > 1.1$ for the update criterion and if $p_{S(m)} < 0.5$ for the score criterion), we will compare the results of the analyses of data sets 1, 9, 13 and 17 for all of the hybrid schemes. The results are presented in Table 8.24.

Using the hybrid schemes with the AI criterion, the AI algorithm failed on the 2nd iteration and the iteration sequence was restarted with an EM type iteration using the initial starting values for data sets 1, 9, 13 and 17. The estimates from this EM type iteration were close enough to the REML solution to then allow successful AI iterations for all of the hybrid schemes. The hybrid schemes with the update and score criteria were able to detect the convergence problems with the AI algorithm on the first iteration and use EM type iterations initially. The number of iterations to convergence were equivalent for the update and score criteria in all cases expect for the PXEM/AI scheme for data set 1 and the EM/AI scheme for data set 13, for which the hybrid schemes with the update criteria converged in one less iteration.
Table 8.24: The iteration results from the analysis of data sets 1, 9, 13 and 17 (of the incomplete block design data sets with 3 replicates) using the various hybrid schemes with the different criteria; 2 internal local iterations were used for the local EM/AI scheme, 1 internal local iteration was used for the local PXEM/AI scheme, a cut-off of 1.1 was used for $\Delta^{(m)}$ for the update criterion and a cut-off of 0.5 was used for $p_{S(m)}$ for the score criterion.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Hybrid scheme</th>
<th>AI criterion</th>
<th>Update criterion</th>
<th>Score criterion</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>EM/AI</td>
<td>13 (1/12)</td>
<td>12 (2/10)</td>
<td>12 (2/10)</td>
</tr>
<tr>
<td></td>
<td>PXEM/AI</td>
<td>10 (1/9)</td>
<td>fail</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td>local EM/AI</td>
<td>fail</td>
<td>fail</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td>local PXEM/AI</td>
<td>fail</td>
<td>fail</td>
<td>fail</td>
</tr>
<tr>
<td>9</td>
<td>EM/AI</td>
<td>10 (1/9)</td>
<td>10 (1/9)</td>
<td>10 (2/8)</td>
</tr>
<tr>
<td></td>
<td>PXEM/AI</td>
<td>9 (1/8)</td>
<td>9 (1/8)</td>
<td>9 (1/8)</td>
</tr>
<tr>
<td></td>
<td>local EM/AI</td>
<td>9 (1/8)</td>
<td>9 (1/8)</td>
<td>9 (1/8)</td>
</tr>
<tr>
<td></td>
<td>local PXEM/AI</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
</tr>
<tr>
<td>13</td>
<td>EM/AI</td>
<td>9 (1/8)</td>
<td>9 (1/8)</td>
<td>10 (2/8)</td>
</tr>
<tr>
<td></td>
<td>PXEM/AI</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
</tr>
<tr>
<td></td>
<td>local EM/AI</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td>8 (2/6)</td>
</tr>
<tr>
<td></td>
<td>local PXEM/AI</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td>17</td>
<td>EM/AI</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
</tr>
<tr>
<td></td>
<td>PXEM/AI</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td></td>
<td>local EM/AI</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
</tr>
<tr>
<td></td>
<td>local PXEM/AI</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
</tr>
</tbody>
</table>

Table 8.25 provides a summary of the number of iterations to convergence for data sets 1, 9, 13 and 17 for the AI, EM and PXEM algorithms, and the hybrid schemes with the update criterion with a cut-off of 1.1 for $\Delta^{(m)}$. The hybrid schemes converged in far fewer iterations than the EM and PXEM algorithms, particularly for data set 13 where the REML estimate for $\sigma^2_u$ was very small (0.03). The performance of the different hybrid schemes was similar, with the PXEM/AI slightly outperforming the EM/AI algorithm. The local schemes failed for data set 1 as they were not able to switch to AI updates and $\sigma^2$ was held fixed at one while $\sigma^2_u$ was updated locally. When the local schemes were successful, the local EM/AI scheme performed just as well as the PXEM/AI algorithm, and the local PXEM/AI algorithm performed just as well or slightly better. The results suggest that the EM/AI or the PXEM/AI algorithm are good algorithms to use to ensure convergence, since the local schemes are not guaranteed to work.

The results of the analyses of the 20 data sets with the design shown in Table 8.6 so far have been based on starting values of $\sigma^2_u(0) = 1$ and $\sigma^2(0) = 1$. We will now look at the results of the analyses of the 20 incomplete block design data sets with 3 replicates using the AI, EM and PXEM algorithms with starting values calculated using Henderson’s Method III. Table 8.26 contains the results of the original analyses and the analyses using HMIII start values, as well as the HMIII start values and the REML estimates of $\sigma^2_u$ and $\sigma^2$. 
Table 8.25: The number of iterations for the analysis of data sets 1,9,13 and 17 (of the incomplete block design data sets with 3 replicates) using the AI, EM, PXEM algorithms as well as the EM/AI, PXEM/AI, local EM/AI and local PXEM/AI schemes with the update criterion with a cut-off of 1.1 for $\Delta^{(m)}$.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Data Set</th>
<th>1</th>
<th>9</th>
<th>13</th>
<th>17</th>
</tr>
</thead>
<tbody>
<tr>
<td>AI</td>
<td></td>
<td>fail</td>
<td>fail</td>
<td>fail</td>
<td>fail</td>
</tr>
<tr>
<td>EM</td>
<td></td>
<td>45</td>
<td>133</td>
<td>5491</td>
<td>71</td>
</tr>
<tr>
<td>PXEM</td>
<td></td>
<td>38</td>
<td>50</td>
<td>301</td>
<td>42</td>
</tr>
<tr>
<td>EM/AI</td>
<td></td>
<td>12 (2/10)</td>
<td>10 (1/9)</td>
<td>9 (1/8)</td>
<td>8 (1/7)</td>
</tr>
<tr>
<td>PXEM/AI</td>
<td></td>
<td>10 (1/9)</td>
<td>9 (1/8)</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
</tr>
<tr>
<td>local EM/AI</td>
<td></td>
<td>fail</td>
<td>9 (1/8)</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
</tr>
<tr>
<td>local PXEM/AI</td>
<td></td>
<td>fail</td>
<td>8 (1/7)</td>
<td>7 (1/6)</td>
<td>8 (1/7)</td>
</tr>
</tbody>
</table>

With the starting values calculated using Henderson’s Method III, the AI algorithm converged for all 20 of the data sets. As expected, the EM and PXEM algorithms converged for all data sets. It is interesting to note that for the data sets for which the EM algorithm took a large number of iterations (> 100) using uninformed starting values, the use of informed starting values did not significantly improve the performance of the EM algorithm. In fact, for data set 13, the number of iterations for the EM algorithm increased from 5491 with uninformed starting values to 7286 with informed starting values. This is the only data set for which the number of iterations increased for the EM algorithm when informed starting values are used. The same pattern is seen for the PXEM algorithm, with the number of iterations decreasing with the use of informed starting values for all data sets except for data set 13. This suggests that good starting values are beneficial for the EM and PXEM algorithms in most cases, but are certainly critical for the AI algorithm. Note that using informed starting values, the AI algorithm converged for the data sets for which it failed to converge using the uninformed starting values (data sets 1, 9, 13 and 17). In each case, it converged in fewer iterations than any of the hybrid schemes.
Table 8.26: The number of iterations to convergence for the AI, EM and PXEM algorithms for the 20 incomplete block design data sets with 3 replicates using the uninformed starting values $\sigma_u^{2(0)}$ and $\sigma^2(0)$ and the starting values calculated via Henderson’s Method III.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>AI</th>
<th>EM</th>
<th>PXEM</th>
<th>AI</th>
<th>EM</th>
<th>PXEM</th>
<th>Starting Values</th>
<th>Starting Values</th>
<th>REML estimates</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>fail</td>
<td>45</td>
<td>38</td>
<td>9</td>
<td>43</td>
<td>33</td>
<td>0.31</td>
<td>0.52</td>
<td>0.31</td>
</tr>
<tr>
<td>2</td>
<td>11</td>
<td>55</td>
<td>48</td>
<td>10</td>
<td>38</td>
<td>43</td>
<td>1.17</td>
<td>0.88</td>
<td>1.36</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>103</td>
<td>45</td>
<td>8</td>
<td>98</td>
<td>42</td>
<td>0.31</td>
<td>1.18</td>
<td>0.35</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>38</td>
<td>32</td>
<td>5</td>
<td>26</td>
<td>27</td>
<td>1.44</td>
<td>1.04</td>
<td>1.41</td>
</tr>
<tr>
<td>5</td>
<td>11</td>
<td>248</td>
<td>82</td>
<td>8</td>
<td>226</td>
<td>74</td>
<td>0.32</td>
<td>1.24</td>
<td>0.28</td>
</tr>
<tr>
<td>6</td>
<td>9</td>
<td>95</td>
<td>56</td>
<td>8</td>
<td>85</td>
<td>49</td>
<td>0.69</td>
<td>1.32</td>
<td>0.66</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>40</td>
<td>39</td>
<td>4</td>
<td>29</td>
<td>27</td>
<td>1.2</td>
<td>0.63</td>
<td>1.16</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>60</td>
<td>41</td>
<td>6</td>
<td>54</td>
<td>37</td>
<td>0.72</td>
<td>1.3</td>
<td>0.75</td>
</tr>
<tr>
<td>9</td>
<td>fail</td>
<td>133</td>
<td>50</td>
<td>7</td>
<td>123</td>
<td>47</td>
<td>0.22</td>
<td>0.94</td>
<td>0.25</td>
</tr>
<tr>
<td>10</td>
<td>8</td>
<td>86</td>
<td>52</td>
<td>7</td>
<td>74</td>
<td>46</td>
<td>0.69</td>
<td>1.28</td>
<td>0.67</td>
</tr>
<tr>
<td>11</td>
<td>10</td>
<td>105</td>
<td>59</td>
<td>8</td>
<td>97</td>
<td>54</td>
<td>0.85</td>
<td>1.61</td>
<td>0.76</td>
</tr>
<tr>
<td>12</td>
<td>6</td>
<td>43</td>
<td>40</td>
<td>4</td>
<td>34</td>
<td>30</td>
<td>1.78</td>
<td>1.62</td>
<td>1.79</td>
</tr>
<tr>
<td>13</td>
<td>fail</td>
<td>5491</td>
<td>301</td>
<td>5</td>
<td>7286</td>
<td>307</td>
<td>0.02</td>
<td>1.53</td>
<td>0.03</td>
</tr>
<tr>
<td>14</td>
<td>7</td>
<td>56</td>
<td>40</td>
<td>7</td>
<td>47</td>
<td>35</td>
<td>0.81</td>
<td>1.34</td>
<td>0.82</td>
</tr>
<tr>
<td>15</td>
<td>10</td>
<td>40</td>
<td>39</td>
<td>6</td>
<td>15</td>
<td>25</td>
<td>2.33</td>
<td>0.52</td>
<td>2.63</td>
</tr>
<tr>
<td>16</td>
<td>7</td>
<td>38</td>
<td>38</td>
<td>4</td>
<td>24</td>
<td>23</td>
<td>2.29</td>
<td>0.6</td>
<td>2.32</td>
</tr>
<tr>
<td>17</td>
<td>fail</td>
<td>71</td>
<td>42</td>
<td>5</td>
<td>56</td>
<td>36</td>
<td>0.39</td>
<td>0.76</td>
<td>0.39</td>
</tr>
<tr>
<td>18</td>
<td>7</td>
<td>39</td>
<td>35</td>
<td>6</td>
<td>32</td>
<td>31</td>
<td>1.02</td>
<td>0.92</td>
<td>0.99</td>
</tr>
<tr>
<td>19</td>
<td>11</td>
<td>99</td>
<td>61</td>
<td>10</td>
<td>91</td>
<td>56</td>
<td>0.84</td>
<td>1.41</td>
<td>0.79</td>
</tr>
<tr>
<td>20</td>
<td>8</td>
<td>35</td>
<td>34</td>
<td>7</td>
<td>27</td>
<td>27</td>
<td>3.26</td>
<td>0.91</td>
<td>2.82</td>
</tr>
</tbody>
</table>

### 8.5 Robust Iterative Scheme

The results have shown that although the PXEM algorithm was an improvement over the EM algorithm, it still fell well short of the AI algorithm when reasonable starting values for the variance parameters were available. However, if the starting values were too far from the REML solution, the AI algorithm failed to converge. For simple variance structures such as the variance components model, it was not difficult to obtain reasonable starting values. However in other cases, reasonable starting values are not always easy to obtain (as we will see in the following chapter). And so we need a scheme that is robust to poor starting values, like the EM and PXEM algorithms, but that is also quick to converge like the AI algorithm.

The EM and PXEM updates over the first few iterations were often closer to the REML solution than the AI estimates, particularly for $\sigma_u^2$. And $r^{(m)}_e$ was often lower over the first few iterations for the EM and PXEM algorithms than for the AI algorithm. So although the EM and PXEM algorithms were slower to converge than the AI algorithm,
they performed well in the first few iterations. Hence we saw that the hybrid schemes performed well in the cases where the AI algorithm failed to converge. The local schemes showed promise but failed to converge for one of the data sets due to the scheme not switching to AI iterations and hence holding $\sigma^2_u$ fixed and only updating $\sigma^2_u$.

The PXEM/AI scheme converged in 1 or 2 fewer iterations than the EM/AI scheme for all cases. However, computationally the EM/AI scheme can be implemented more efficiently than the PXEM/AI algorithm. An iteration of the EM/AI or PXEM/AI scheme starts by calculating the AI update and then checks to see if the AI update is within the parameter space (i.e. the variance parameter estimates result in a positive definite AI matrix) or to see if it looks like the AI update is heading in the right direction. If either of these conditions are not met, then an EM or PXEM update is calculated. To calculate the AI update, we need to calculate the score vector. Once we have calculated the score for $\sigma^2_u$, as given in (8.2), we have all the elements we need to calculate the EM update for $\sigma^2_u$, as given in (8.5); namely $\tilde{u}$ and $\text{tr}(CZZ)$. However, to calculate the PXEM update for $\sigma^2_u$, as given in (8.7), we also need to calculate $\text{tr}(C^{XZ})$ which can be computationally expensive (Cullis et al., 2004). Note that the score for the EM update for $\sigma^2$, as given in (8.6), can be calculated in terms of $C^{ZZ}$, allowing us to calculate the EM update without having to calculate $C^{-1}$. So for our robust scheme we will use the EM/AI scheme as it performs almost as well as the PXEM/AI scheme yet is computationally simpler to implement than the PXEM/AI scheme.

In our final analyses of the data sets for which that AI algorithm did not converge (as presented in Tables 8.15 and 8.25) we chose to use the update criterion with a cut-off of 1.1. We chose to use the update criterion in preference to the score criterion as we were able to find a cut-off for the update criterion that invoked EM iterations only for the data sets for which AI failed to converge while detecting the convergence problems before they occurred. A cut-off of 0.5 for the score criterion invoked an EM iteration before the AI algorithm failed and hence detected convergence problems before they occurred. The results in Tables 8.14 and 8.24 suggest that a cut-off of 0.5 for the score criterion performed just as well as a cut-off of 1.1 for the update criterion. However the score criterion with a cut-off of 0.5 invoked EM iterations for a number of the data sets for which the AI algorithm successfully converged.

The update criterion weights all of the variance parameters equally, while the score criterion incorporates a weighting scheme. At the REML solution, $I_A(\kappa^{(m)})^{-1}$ provides an estimate of the covariance matrix for the variance parameters. Thus $S^{(m)} = U(\kappa^{(m)})I_A(\kappa^{(m)})^{-1}U(\kappa^{(m)})$ incorporates a weighting scheme that takes into account the precision with which the variance parameter estimates are known. Changes in variance parameter estimates with little precision are given more weight (Belsley, 1980). The convergence problems encountered when using the AI algorithm are often related to the parameters in the random effects variance matrix $G$, rather than the error variance matrix $R$ (Cullis et al., 2004). These parameters typically have larger variances and so we have decided to use the score criterion for our final robust scheme as it gives more weight to the variance parameters with larger variances. We could just use a cut-off for $S^{(m)}$ as is often done when $S^{(m)}$ is used as a convergence criterion (Fletcher, 1987, Dennis and Schnabel, 1996, Demidenko, 2004). However comparing $S^{(m)}$ with a reference distribution of a chi-squared distribution with $k$ degrees of freedom (where $k$ is the number of vari-
Our robust scheme will use an EM iteration if the AI update is outside of the parameter space or if $p_{S(m)} < 0.5$. So far we have defined the parameter space as all values of the variance parameters such that $\mathcal{I}_A(\kappa^{(m)})$ is positive definite. For the cases where the AI algorithm failed, it did so because the AI matrix was not positive definite on the second iteration. The AI matrix in the second iteration was calculated using the variance parameter estimates from the first iteration. In each case, $\sigma_u^2(1)$ was negative and the AI algorithm could have been terminated at the first iteration if we defined the parameter space as all values of the variance parameters such that $\sigma_u^2 > 0$ and $\sigma^2 > 0$. Note that by doing so, the AI algorithm would have been deemed to have failed for data set 3 of the simulated incomplete block design data sets with 3 replicates, since $\sigma_u^2(1) < 0$, even though it did converge. However, as we will see, invoking an EM iteration initially results in fewer iterations than the AI algorithm alone. To decide on an optimal iterative strategy, we will compare the results obtained using the two different parameter space definitions.

As we have already discussed, once we have calculated an AI iteration, we have all the pieces we need to calculate an EM update. If an EM iteration is required due to the AI update being outside of the parameter space or due to $p_{S(m)} < 0.5$, a computationally convenient approach is to calculate the EM update using the current variance parameter estimates. This approach will work well if an EM update is invoked on the first iteration and hence the EM update will be calculated using the initial starting values. However, if an EM update is not invoked until the second iteration or later, it will be calculated using the variance parameter estimates from the previous AI iteration, which may have already been heading in the wrong direction. Hence, the first time an EM update is invoked, a better approach might be to use the initial starting values to calculate the EM update. And if another EM update is invoked after the iteration process has been restarted using an EM update, use the variance parameter estimates from the previous EM iteration to make use of the progress that was made with the earlier EM update. To decide on an optimal iterative strategy, we will compare the results obtained using the two approaches; calculating an EM update using the current variance parameter estimates, or resetting to the initial starting values or the estimates from the previous EM iteration.

To determine an optimal iterative strategy we will look at 4 different variations of the EM/AI scheme with the score criterion with a cut-off of 0.5 for $p_{S(m)}$. We will compare the results obtained using 2 different definitions of the parameter space; $\mathcal{I}_A(\kappa)$ positive definite (denoted by $\mathcal{I}_A(\kappa) > 0$) versus $\sigma_u^2 > 0$ and $\sigma^2 > 0$. We will also compare the results obtained by resetting to the initial starting values or the estimates from the previous EM iteration to calculate an EM update (referred to as ”reset”) versus using the current estimates (referred to as ”current”). We will also compare the results of these robust algorithms to the current methods used in popular software. ASReml and the REML directive in GenStat use the AI algorithm, and so we will compare the performance of our robust schemes to the AI algorithm. The lmer function in the lme4 package in R uses 15 EM iterations and then switches to Newton Raphson iterations. We use the AI algorithm in preference to the Newton Raphson algorithm, and so will compare the performance of our robust schemes to a scheme that uses 15 EM iterations and then switches to...
AI iterations (referred to as "lmer"). Finally, we will also compare the performance of our robust schemes to the AI algorithm with informed starting values obtained using Henderson’s Method III. Table 8.27 presents the number of iterations to convergence for each of these various algorithms for the lamb weight data and the simulated incomplete block design data sets; 20 data sets with 9 replicates of each of 12 treatments, and 20 data sets with 3 replicates of each of 12 treatments.

For the data sets for which EM iterations were invoked, the first EM iteration was invoked in the first iteration and subsequent EM iterations were invoked directly following EM iterations. Hence the robust schemes that calculated the EM iterations using the current estimates performed just as well as those that reset to the initial starting values for the first EM iteration or to the estimates from the previous EM iteration for subsequent EM iterations. This was due to the ability of the score criterion to detect potential convergence problems with the AI algorithm.

For the robust schemes with the parameter space defined as all values of $\sigma^2_u$ and $\sigma^2$ such that $I_A(\kappa^{(m)}) > 0$, all EM iterations were invoked due to $p_{S(m)} < 0.5$ rather than due to $I_A(\kappa^{(m)}) > 0$. When the parameter space was defined as all values of $\sigma^2_u > 0$ and $\sigma^2 > 0$, all of the EM iterations used were invoked due to $p_{S(m)} < 0.5$ except for those invoked for data set 8 of the incomplete block design data sets with 9 replicates, and data sets 1, 3, 9, 13 and 17 of the incomplete block design data sets with 3 replicates. For these data sets the EM iterations were invoked due to AI estimates of $\sigma^2$ being negative.

The 4 robust schemes converged in the same number and type of iterations for each data set except for data set 13 of the incomplete block design data sets with 3 replicates. In this case, defining the parameter space as all values of $\sigma^2_u$ and $\sigma^2$ such that $I_A(\kappa^{(m)}) > 0$ resulted in the use of 2 EM iterations due to $p_{S(m)} < 0.5$ for the first 2 attempted AI iterations. These two EM iterations were enough to obtain variance parameter estimates close enough to the solution to allow AI to converge. However, note that the first two AI iterations (after 2 EM iterations) produced negative estimate of $\sigma^2_u$. Requiring $\sigma^2_u^{(m)} > 0$ and $\sigma^2(m) > 0$ resulted in 10 EM iterations, each invoked due to the AI estimate of $\sigma^2_u$ being negative. This approach resulted in a total of 16 iterations compared to 9 iterations when the definition of the parameter space was relaxed to all values of $\sigma^2_u$ and $\sigma^2$ such that $I_A(\kappa^{(m)}) > 0$.

Defining the parameter space as all values of $\sigma^2_u > 0$ and $\sigma^2 > 0$ is a more intuitive approach as it based on the properties of the model rather than the algorithm. Requiring $\sigma^2_u^{(m)} > 0$ and $\sigma^2(m) > 0$ resulted in a few more iterations for one of the 41 data sets, but performed just as well as the more relaxed definition of the parameter space ($\sigma^2_u$ and $\sigma^2$ such that $I_A(\kappa^{(m)}) > 0$) in all other cases. Hence our preferred approach is to define the parameter space as all values of $\sigma^2_u > 0$ and $\sigma^2 > 0$. If an EM iteration was invoked, it was invoked in the first iteration and subsequent EM iterations were invoked directly following EM iterations, suggesting that we can take the computationally simpler approach and use the current estimates to calculate the EM iteration, rather than resetting to the initial starting values for the first EM iteration or to the estimates from the previous EM iteration for subsequent EM iterations. Hence our preferred robust scheme uses AI iterations unless the AI iteration results in $\sigma^2_u^{(m)} < 0$ or $\sigma^2(m) < 0$, or if $p_{S(m)} < 0.5$, in which case an EM iteration is done in place of the AI iteration.
Table 8.27: The number of iterations to convergence for the lamb weight data, the 20 incomplete block design data sets with 9 replicates and the 20 incomplete block design data sets with 3 replicates. The results are for the AI algorithm, lmer scheme, 4 different versions of the robust hybrid EM/AI schemes with the score criterion with a cut-off of 0.5 for $p_{S(m)}$, all using the uninformed starting values, and the AI algorithm using informed starting values obtained using Henderson’s Method III. The results are presented in the form “25 (15/10)” where 25 is the total number of iterations, 15 is the number of EM iterations and 10 is the number of AI iterations.

<table>
<thead>
<tr>
<th>data set</th>
<th>AI</th>
<th>lmer</th>
<th>reset</th>
<th>current</th>
<th>reset</th>
<th>current</th>
<th>AI HMI</th>
<th>III</th>
</tr>
</thead>
<tbody>
<tr>
<td>lamb</td>
<td>13</td>
<td>25</td>
<td>(15/10)</td>
<td>12</td>
<td>(2/10)</td>
<td>12</td>
<td>(2/10)</td>
<td>12</td>
</tr>
<tr>
<td>r=9</td>
<td>2</td>
<td>17</td>
<td>(15/2)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>17</td>
<td>(15/2)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>17</td>
<td>(15/2)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>17</td>
<td>(15/2)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>17</td>
<td>(15/2)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td>13</td>
<td>(13/0)</td>
<td>5</td>
<td>(0/5)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>13</td>
<td>(13/0)</td>
<td>5</td>
<td>(0/5)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>13</td>
<td>(13/0)</td>
<td>5</td>
<td>(0/5)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>16</td>
<td>(15/1)</td>
<td>5</td>
<td>(0/5)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>17</td>
<td>(15/2)</td>
<td>7</td>
<td>(0/7)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>12</td>
<td>(12/0)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>16</td>
<td>(15/1)</td>
<td>6</td>
<td>(0/6)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>13</td>
<td>(13/0)</td>
<td>6</td>
<td>(0/6)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>12</td>
<td>(12/0)</td>
<td>5</td>
<td>(0/5)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>13</td>
<td>(13/0)</td>
<td>5</td>
<td>(0/5)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>17</td>
<td>15</td>
<td>(15/0)</td>
<td>5</td>
<td>(0/5)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>18</td>
<td>13</td>
<td>(13/0)</td>
<td>5</td>
<td>(0/5)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>19</td>
<td>12</td>
<td>(12/0)</td>
<td>5</td>
<td>(0/5)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>13</td>
<td>(13/0)</td>
<td>5</td>
<td>(0/5)</td>
<td>6</td>
<td>(1/5)</td>
<td>6</td>
</tr>
<tr>
<td>IB</td>
<td>1</td>
<td>fail</td>
<td>22</td>
<td>(15/7)</td>
<td>11</td>
<td>(0/11)</td>
<td>11</td>
<td>(0/11)</td>
</tr>
<tr>
<td>r=3</td>
<td>2</td>
<td>23</td>
<td>(15/8)</td>
<td>11</td>
<td>(0/11)</td>
<td>11</td>
<td>(0/11)</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>23</td>
<td>(15/8)</td>
<td>10</td>
<td>(0/11)</td>
<td>11</td>
<td>(0/11)</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>19</td>
<td>(15/4)</td>
<td>6</td>
<td>(0/6)</td>
<td>6</td>
<td>(0/6)</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>24</td>
<td>(15/9)</td>
<td>11</td>
<td>(0/11)</td>
<td>11</td>
<td>(0/11)</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>22</td>
<td>(15/7)</td>
<td>9</td>
<td>(0/9)</td>
<td>9</td>
<td>(0/9)</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>19</td>
<td>(15/4)</td>
<td>7</td>
<td>(0/9)</td>
<td>9</td>
<td>(0/9)</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>21</td>
<td>(15/6)</td>
<td>7</td>
<td>(0/9)</td>
<td>7</td>
<td>(0/9)</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>fail</td>
<td>22</td>
<td>(15/7)</td>
<td>10</td>
<td>(0/9)</td>
<td>7</td>
<td>(0/9)</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>21</td>
<td>(15/6)</td>
<td>8</td>
<td>(0/9)</td>
<td>8</td>
<td>(0/9)</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>23</td>
<td>(15/8)</td>
<td>8</td>
<td>(0/9)</td>
<td>8</td>
<td>(0/9)</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>19</td>
<td>(15/4)</td>
<td>8</td>
<td>(0/9)</td>
<td>8</td>
<td>(0/9)</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>fail</td>
<td>21</td>
<td>(15/6)</td>
<td>9</td>
<td>(0/9)</td>
<td>9</td>
<td>(0/9)</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>21</td>
<td>(15/6)</td>
<td>7</td>
<td>(0/9)</td>
<td>9</td>
<td>(0/9)</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>20</td>
<td>(15/6)</td>
<td>7</td>
<td>(0/9)</td>
<td>9</td>
<td>(0/9)</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>18</td>
<td>(15/3)</td>
<td>8</td>
<td>(0/9)</td>
<td>9</td>
<td>(0/9)</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>17</td>
<td>fail</td>
<td>20</td>
<td>(15/6)</td>
<td>7</td>
<td>(0/9)</td>
<td>9</td>
<td>(0/9)</td>
</tr>
<tr>
<td></td>
<td>18</td>
<td>20</td>
<td>(15/6)</td>
<td>7</td>
<td>(0/9)</td>
<td>9</td>
<td>(0/9)</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>19</td>
<td>24</td>
<td>(15/9)</td>
<td>11</td>
<td>(0/11)</td>
<td>11</td>
<td>(0/11)</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>20</td>
<td>(15/6)</td>
<td>8</td>
<td>(0/9)</td>
<td>9</td>
<td>(0/9)</td>
<td>9</td>
</tr>
</tbody>
</table>

\[ I_A(k(m)) > 0 \quad \sigma_n^2 > 0 \quad \sigma_u^2 > 0 \]
Now we will look at how this robust scheme compared to popular current approaches. The robust scheme converged in the same number of iterations or fewer iterations than the AI algorithm for the cases where the AI algorithm converged, except for data set 16 of the incomplete block design date sets with 3 replicates, for which the robust scheme converged in 1 more iteration than the AI algorithm. For the cases where the AI algorithm failed to converge, the robust scheme successfully converged in all cases. Therefore the results show that our robust algorithm is an improvement over the AI algorithm used in Genstat and ASReml.

In almost all cases, our robust scheme converged in less than half of the number of iterations for the lmer approach. For half of the data sets, EM iterations were not required. In the cases where EM iterations were invoked, a single EM iteration was all that was needed in most cases. Hence our approach of only using EM iterations if we are too far from the REML solution and then switching back to AI iterations is an improvement on the lmer approach which uses 15 EM iterations and then switches to AI iterations.

Finally we compare the results obtained using our robust scheme to those obtained using the AI algorithm with informed starting values. The AI algorithm with starting values obtained using Henderson’s Method III, outperformed our robust scheme for all data sets except for two data sets for which the two schemes converged in the same number of iterations. However, reasonable starting values are not always easy to obtain for more complex variance models and hence we need a scheme that is robust to poor starting values. The results for the analyses of the variance components data sets show that the difference in the number of iterations to convergence was only 2 or 3 iterations for most of the data sets. Hence we have been able to find a scheme that while being robust, also demonstrates fast convergence.
Chapter 9

Unstructured G Model

9.1 Introduction

In this chapter we investigate and compare the performance of the various iterative schemes for a linear mixed model with a more complicated random effects variance matrix, a direct product of an unstructured variance matrix and the identity matrix, that can be used for random coefficient regression and the analysis of data from multi-environment plant variety trials and multi-trait animal breeding trials. Random coefficient regression models (Laird and Ware, 1982, Longford, 1993, Diggle et al., 2002) can be used in situations where there are repeated measurements taken on $m$ subjects or experimental units over $q$ times or experimental conditions. A polynomial response curve, defined by $p$ regression coefficients, is fitted to the repeated measurements for each subject. The response curves all have the same form but vary randomly among individuals. Linear mixed models with an unstructured variance model for the random effects can be used to analyse multi-environment plant variety trial data (Smith et al., 2001b, 2005) involving $r$ replicates of $m$ varieties grown in $p$ environments or multi-trait animal breeding data (Thompson, 1979, Thompson and Meyer, 1986, Meyer, 1989) involving $m$ sires and $p$ traits.

We present the form of the updating equations for the variance parameters for the AI, EM and PXEM algorithms and the local EM and local PXEM schemes. The performance of the various iterative schemes is compared for random coefficient models using data from Potthoff and Roy (1964) and Vonesh and Carter (1992), as well as simulated data, and for simulated variety by environment data. We present the form of the updating equations for the variance parameters for the AI, EM and PXEM algorithms and the local EM and local PXEM schemes. For the data sets for which the AI algorithm does not converge, we investigate the use of the hybrid schemes, including selection of the number of internal local iterations for the local schemes and the cut-offs for the update and score criteria for the hybrid schemes. We will then use these results to develop an algorithm that is both robust and converges rapidly.
9.2 Unstructured G Model and Results

We will consider a model with \( p \) regression coefficients, environments or traits for each of \( m \) subjects, varieties or sires. We assume that the data are ordered observations within individuals, replicates of varieties within environments or offspring within sires within traits. If \( y \) is the \( n \times 1 \) vector of observations then the model is given by

\[
y = X\tau + Zu + e
\]

where \( \tau \) is a \( t \times 1 \) vector of fixed effects with associated \( n \times t \) full rank design matrix \( X \), \( u \) is the \( mp \times 1 \) vector of random effects for \( p \) coefficients, environments or traits for each of the \( m \) subjects, varieties or sires, with associated \( n \times mp \) design matrix \( Z \), and \( e \) is the \( n \times 1 \) vector of residuals.

We assume that the joint distribution of the random effects and residuals is given by

\[
\begin{bmatrix} u \\ e \end{bmatrix} \sim N \left( \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} G & 0 \\ 0 & R \end{bmatrix} \right)
\]

The \( mp \times 1 \) vector of random effects \( u \) can be represented as a two-dimensional array of effects, that is an \( m \times p \) matrix \( U \) given by

\[
U = \begin{bmatrix}
u_{11} & u_{12} & \cdots & u_{1p} \\
u_{21} & u_{22} & \cdots & u_{2p} \\
\vdots & \vdots & \ddots & \vdots \\
u_{m1} & u_{m2} & \cdots & u_{mp}
\end{bmatrix}
\]
such that \( u = \text{vec}(U) \). Hence \( u \) is partitioned into \( p \) vectors of length \( m \times 1 \), that is \( u' = (u'_1, u'_2, \ldots, u'_p) \) where \( u'_i = (u_{i1}, u_{i2}, \ldots, u_{im}) \). We assume that the variance of \( u \) is given by \( G = G_p \otimes I_m \) where \( G_p \) is a \( p \times p \) symmetric unstructured variance matrix given by

\[
G_p = \begin{bmatrix}
\gamma_{11} & \gamma_{12} & \cdots & \gamma_{1p} \\
\gamma_{12} & \gamma_{22} & \cdots & \gamma_{2p} \\
\vdots & \vdots & \ddots & \vdots \\
\gamma_{1p} & \gamma_{2p} & \cdots & \gamma_{pp}
\end{bmatrix}
\]

and hence

\[
G = \begin{bmatrix}
\gamma_{11}I_m & \gamma_{12}I_m & \cdots & \gamma_{1p}I_m \\
\gamma_{12}I_m & \gamma_{22}I_m & \cdots & \gamma_{2p}I_m \\
\vdots & \vdots & \ddots & \vdots \\
\gamma_{1p}I_m & \gamma_{2p}I_m & \cdots & \gamma_{pp}I_m
\end{bmatrix}
\]

\( G_p \) is a function of \( \gamma \) where \( \gamma = \text{vech}(G_p) = (\gamma_{11}, \gamma_{12}, \ldots, \gamma_{1p}, \gamma_{22}, \ldots, \gamma_{2p}, \ldots, \gamma_{pp})' \).
Note that Smith et al. (2001b), Smith et al. (2005) and Kelly et al. (2007) describe a more general form of $G = G_e \otimes G_v$ in the context of multi-environment plant variety trials, where $G_e$ and $G_v$ are $p \times p$ and $m \times m$ symmetric positive definite matrices for environments and varieties, respectively. Following Smith et al. (2001b), we have assumed that $G_v = I_m$ and hence $G = G_p \otimes I_m$ where the matrix $G_e$ is referred to as the genetic variance matrix with the diagonal elements representing genetic variances for individual environments and the off-diagonal elements representing genetic covariances between pairs of environments. An alternative model is to use $G_v = A$ where $A$ is a known relationship matrix reflecting the pedigrees or marker genotypes of the varieties. A similar model can be used in the context of multi-trait animal models, $G = G_p \otimes A$ where $G_p$ is the genetic variance matrix and $A$ is the relationship matrix of the sires in the system (Henderson, 1976, Meyer, 1989, 1991).

We assume that $R = \sigma^2 I_n$, that is the residual errors have constant variance and are uncorrelated. This assumption does not always hold, particularly in the analysis of multi-environment plant variety trial and multi-trait animal breeding data. For example, Smith et al. (2001b) use $R = \Sigma_e \otimes \Sigma_r$ to account for the spatial correlation induced by the underlying row and column structure in multi-environment plant variety trials.

The variance parameters to be estimated are $\kappa = (\gamma', \sigma^2)$.

The distribution of $y$ is given by

$$y \sim N(X\tau, H)$$

where $H = Z(G_p \otimes I_m)Z' + \sigma^2 I_n$.

The REML log-likelihood for this model is given by

$$\ell_R(\gamma, \sigma^2; y_2) = -\frac{1}{2} (\log |G_p \otimes I_m| + \log |\sigma^2 I_n| + \log |C| + y'Py)$$

Using Result A.13 we find that $|G_p \otimes I_m| = |G_p|^m |I_m|^p = |G_p|^m$ and hence

$$\log |G_p \otimes I_m| = m \log |G_p|$$

and noting that $|\sigma^2 I_n| = (\sigma^2)^n$ gives

$$\log |\sigma^2 I_n| = n \log \sigma^2$$

This allows us to write the REML log-likelihood as

$$\ell_R(\gamma, \sigma^2; y_2) = -\frac{1}{2} \left( m \log |G_p| + n \log \sigma^2 + \log |C| + y'Py \right)$$

The REML score for $\gamma_{ij}$ for the general linear mixed model is given by

$$U(\gamma_{ij}) = -\frac{1}{2} \left( \text{tr}(G^{-1} \tilde{G}_{ij}) - \text{tr}(G^{-1} \tilde{G}_{ij} G^{-1} C Z Z') - \tilde{u}' G^{-1} \tilde{G}_{ij} G^{-1} \tilde{u} \right)$$

For the unstructured $G$ model in (9.1), $G^{-1} = G_p^{-1} \otimes I_m$ and $\tilde{G}_{ij} = \tilde{G}_{pij} \otimes I_m$ where $\tilde{G}_{pij} = \frac{\partial G_e}{\partial \gamma_{ij}}$. Therefore

$$G^{-1} \tilde{G}_{ij} = (G_p^{-1} \otimes I_m)(\tilde{G}_{pij} \otimes I_m) = (G_p^{-1} \tilde{G}_{pij} \otimes I_m)$$
\[ \text{tr}(G^{-1}\dot{G}_{p_{ij}}) = (G^{-1}_{p} \dot{G}_{p_{ij}} \otimes I_{m}) \]
\[ = m\text{tr}(G^{-1}_{p} \dot{G}_{p_{ij}}) \]

and

\[ G^{-1}\dot{G}_{ij}G^{-1} = (G^{-1}_{p} \otimes I_{m})(\dot{G}_{p_{ij}} \otimes I_{m})(G^{-1}_{p}) \]
\[ = (G^{-1}_{p} \dot{G}_{p_{ij}} G^{-1}_{p} \otimes I_{m}) \]

Hence the score for \( \gamma_{ij} \) for this model is given by as

\[ U(\gamma_{ij}) = -\frac{1}{2} \left( m\text{tr}(G^{-1}_{p} \dot{G}_{p_{ij}}) - \text{tr}((G^{-1}_{p} \dot{G}_{p_{ij}} G^{-1}_{p} \otimes I_{m})C_{ZZ}) - \hat{u}'(G^{-1}_{p} \dot{G}_{p_{ij}} G^{-1}_{p} \otimes I_{m})\hat{u} \right) \]  \hspace{1cm} (9.2)

To obtain an update for \( \gamma_{ij} \) we need to equate the above score for zero and solve for \( \gamma_{ij} \).

An algebraic solution cannot be found and an iterative scheme is required.

The REML score for \( \sigma^2 \) is given by

\[ U(\sigma^2) = -\frac{1}{2} \left( \frac{n - t - mp + \text{tr}(C_{ZZ}(G^{-1}_{p} \otimes I_{m}))}{\sigma^2} - \frac{\hat{e}'\hat{e}}{\sigma^4} \right) \]

Note that \( C_{ZZ} \) and \( \hat{e} \) are functions of \( \sigma^2 \) and hence an algebraic solution for \( \sigma^2 \) cannot be found and an iterative scheme is required.

The observed information matrix is given by

\[ I_{O} = \begin{bmatrix} I_{O}(\gamma, \gamma) & I_{O}(\gamma, \sigma^2) \\ I_{O}(\sigma^2, \gamma) & I_{O}(\sigma^2, \sigma^2) \end{bmatrix} \]

where the elements are given by

\[ I_{O}(\gamma_{ij}, \gamma_{kl}) = y'PZG_{ij}Z'PZ\dot{G}_{kl}Z'Py - \frac{1}{2} \text{tr} \left( PZ\dot{G}_{ij}Z'PZ\dot{G}_{kl}Z' \right) \]

\[ I_{O}(\gamma_{ij}, \sigma^2) = y'PZG_{ij}Z'P^2y - \frac{1}{2} \text{tr} \left( PZ\dot{G}_{ij}Z'P \right) \]

\[ I_{O}(\sigma^2, \sigma^2) = y'P^3y - \frac{1}{2} \text{tr} \left( P^2 \right) \]
9.2.1 AI Algorithm

Given an initial estimate of $\kappa$, $\kappa^{(m)'} = (\gamma^{(m)'} , \sigma^2(m))$, an update of $\gamma$ and $\sigma^2$ using the AI algorithm is given by

$$
\begin{bmatrix}
\gamma^{(m+1)} \\
\sigma^2(m+1)
\end{bmatrix}
= 
\begin{bmatrix}
\gamma^{(m)} \\
\sigma^2(m)
\end{bmatrix}
+ \mathcal{I}_A^{(m)-1}
\begin{bmatrix}
U(\gamma^{(m)}) \\
U(\sigma^2(m))
\end{bmatrix}
$$

where $\mathcal{I}_A$ is partitioned as

$$
\mathcal{I}_A = 
\begin{bmatrix}
\mathcal{I}_A(\gamma, \gamma) & \mathcal{I}_A(\gamma, \sigma^2) \\
\mathcal{I}_A(\sigma^2, \gamma) & \mathcal{I}_A(\sigma^2, \sigma^2)
\end{bmatrix}
$$

and given by

$$
\mathcal{I}_A = \frac{1}{2} Q'PQ
$$

For the unstructured $G$ model $Q = [ q_{\gamma_1} \cdots q_{\gamma_p} q_{\sigma^2} ]$ where $q_{\gamma_i}$ is the working variable for $\gamma_{ij}$ and $q_{\sigma^2}$ is the working variable for $\sigma^2$.

Recall from (3.11) in Chapter 3 that the working variable for $\kappa_i$ is $q_i = \bar{H}_iPy$ where $\bar{H}_i = \frac{\partial H}{\partial \kappa_i}$. For the unstructured $G$ model $H = Z(G_p \otimes I_m)Z' + \sigma^2 I_m$ and hence

$$
\frac{\partial H}{\partial \gamma_{ij}} = Z(\hat{G}_{pij} \otimes I_m)Z'
$$

$$
\frac{\partial H}{\partial \sigma^2} = I_n
$$

Recall from Chapter 2 that $\tilde{e} = RPy$ and $\tilde{u} = GZ'Py$ for the general linear mixed model. Note that for the unstructured $G$ model, $\tilde{u} = (G_p \otimes I_m)Z'Py$ and $\tilde{e} = \sigma^2 Py$. This allows us to write the working variable for $\gamma_{ij}$ as

$$
q_{\gamma_{ij}} = Z(\hat{G}_{pij} \otimes I_m)Z'Py = Z(\hat{G}_{pij} \otimes I_m)(G_p^{-1} \otimes I_m)\tilde{u}
$$

$$
= Z(\hat{G}_{pij} G_p^{-1} \otimes I_m)\tilde{u}
$$

(9.3)

and the working variable for $\sigma^2$ as

$$
\begin{align*}
q_{\sigma^2} &= Py \\
&= \frac{\tilde{e}}{\sigma^2}
\end{align*}
$$

The convergence rate matrix of the AI algorithm is given by

$$
M'_{AI}(\hat{\kappa}) = I - \mathcal{I}_A(\hat{\kappa})^{-1}\mathcal{I}_O(\hat{\kappa})
$$

and the rate of convergence rate is given by the largest eigenvalue of $M'_{AI}(\hat{\kappa})$. 
9.2.2 EM Algorithm

The M step for $\gamma_{ij}$ is given by

$$\frac{\partial E_{EM}[f_c(\kappa)]^{(m)}}{\partial \gamma_{ij}} = \frac{\partial E_{EM}[f_c(\gamma)]^{(m)}}{\partial \gamma_{ij}}$$

$$= -\frac{1}{2} \left( \text{tr}(G^{-1}\tilde{G}_{ij}) - \text{tr}(CZZ^{(m)}G^{-1}\tilde{G}_{ij}G^{-1}) - \tilde{u}^{(m)'}G^{-1}\tilde{G}_{ij}G^{-1}\tilde{u}^{(m)} \right)$$

For the unstructured $G$ model, $G^{-1} = G_p^{-1} \otimes I_m$ and $\tilde{G}_{ij} = \tilde{G}_{p_{ij}} \otimes I_m$, therefore the first term in the M step for $\gamma_{ij}$ is given by

$$\text{tr}(G^{-1}\tilde{G}_{ij}) = \text{tr}\left((G_p^{-1} \otimes I_m)(\tilde{G}_{p_{ij}} \otimes I_m)\right)$$

$$= \text{tr}(G_p^{-1}\tilde{G}_{p_{ij}} \otimes I_m)$$

$$= n\text{tr}(G_p^{-1}\tilde{G}_{p_{ij}})$$

, the second term is given by

$$\text{tr}(CZZ^{(m)}G^{-1}\tilde{G}_{ij}G^{-1}) = \text{tr}\left(CZZ^{(m)}(G_p^{-1} \otimes I_m)(\tilde{G}_{p_{ij}} \otimes I_m)(G_p^{-1} \otimes I_m)\right)$$

$$= \text{tr}\left(CZZ^{(m)}(G_p^{-1}\tilde{G}_{p_{ij}}G_p^{-1} \otimes I_m)\right)$$

and the third term is given by

$$\tilde{u}^{(m)'}G^{-1}\tilde{G}_{ij}G^{-1}\tilde{u}^{(m)} = \text{vec}(\tilde{U}^{(m)})'(G_p^{-1}\tilde{G}_{p_{ij}}G_p^{-1} \otimes I_m)\text{vec}(\tilde{U}^{(m)})$$

$$= \text{tr}(G_p^{-1}\tilde{G}_{p_{ij}}G_p^{-1}\tilde{u}^{(m)'}\tilde{u}^{(m)})$$

$$= \text{tr}(G_p^{-1}\tilde{G}_{p_{ij}}G_p^{-1}\tilde{\Delta}^{(m)})$$

using Result A.13, where $\tilde{U}$ is the $m \times p$ matrix such that $\text{vec}(\tilde{U}) = \tilde{u}$, and $\tilde{\Delta}^{(m)} = \tilde{U}^{(m)'}\tilde{u}^{(m)}$.

We can now write the M step for $\gamma_{ij}$ as

$$\frac{\partial E_{EM}[f_c(\kappa)]^{(m)}}{\partial \gamma_{ij}} = -\frac{1}{2} \left( n\text{tr}(G_p^{-1}\tilde{G}_{p_{ij}}) - \text{tr}(CZZ^{(m)}(G_p^{-1}\tilde{G}_{p_{ij}}G_p^{-1} \otimes I_m)) - \text{tr}(G_p^{-1}\tilde{G}_{p_{ij}}G_p^{-1}\tilde{\Delta}^{(m)}) \right)$$

(9.4)

To obtain an update for $\gamma_{ij}$ we need to further simplify (9.4). $G_p$ is a symmetric matrix and hence $G_p^{-1}$ is also a symmetric matrix. We denote the unique elements of $G_p$ by $\gamma_{ij}$, $i \leq j = 1, \ldots, p$ and the unique elements of $G_p^{-1}$ by $\gamma_{ij}$, $i \leq j = 1, \ldots, p$.

Firstly consider the term $\text{tr}(G_p^{-1}\tilde{G}_{p_{ij}})$. When $i = j$ the matrix $\tilde{G}_{p_{ij}}$ is a matrix of zeros except for a one in the $(i,i)$th position. Hence the matrix $G_p^{-1}\tilde{G}_{p_{ij}}$ is a matrix of zeros except for the $i$th column which is the $i$th column of $G_p^{-1}$. Thus

$$\text{tr}(G_p^{-1}\tilde{G}_{p_{ij}}) = \gamma_{ii}$$
When \( i \neq j \) the matrix \( G_{pij} \) is a matrix of zeros except for ones in the \((i, j)\)th and \((j, i)\)th positions. Hence the matrix \( G_p^{-1} G_{pij} \) is a matrix of zeros except for the \(i\)th and \(j\)th columns which are the \(j\)th and \(i\)th columns of \( G_p^{-1} \) respectively. Thus

\[
\text{tr}(G_p^{-1} G_{pij}) = \gamma^{ij} + \gamma^{ji} = 2\gamma^{ij}
\]
due to the symmetry of \( G_p^{-1} \). Therefore

\[
\text{tr}(G_p^{-1} G_{pij}) = \begin{cases} \gamma^{ii}, & i = j \\ 2\gamma^{ij}, & i \neq j \end{cases} \tag{9.5}
\]

We then consider

\[
\text{tr}(G_p^{-1} G_{pij} G_p^{-1} \Delta^{(m)}) = \text{tr}(G_p^{-1} \Delta^{(m)} G_p^{-1} G_{pij}) = \text{tr}(\Delta^{(m)} G_{pij})
\]

where \( \Delta^{(m)} = G_p^{-1} \Delta G_p^{-1} = \{\delta^{(m)}_{ij}\} \). When \( i = j \) the matrix \( \Delta^{(m)} G_{pij} \) is a matrix of zeros except for the \(i\)th column which is the \(i\)th column of \( \Delta^{(m)} \). Thus

\[
\text{tr}(\Delta^{(m)} G_{pij}) = \delta^{(m)}_{ii}
\]

When \( i \neq j \) the matrix \( \Delta^{(m)} G_{pij} \) is a matrix of zeros except for the \(i\)th and \(j\)th columns which are the \(j\)th and \(i\)th columns of \( \Delta^{(m)} \) respectively. Thus

\[
\text{tr}(\Delta^{(m)} G_{pij}) = \delta^{(m)}_{ij} + \delta^{(m)}_{ji} = 2\delta^{(m)}_{ij}
\]
due to the symmetry of \( \Delta^{(m)} \). Therefore

\[
\text{tr}(G_p^{-1} G_{pij} G_p^{-1} \Delta^{(m)}) = \begin{cases} \delta^{(m)}_{ii}, & i = j \\ 2\delta^{(m)}_{ij}, & i \neq j \end{cases} \tag{9.6}
\]

To simplify the remaining term in (9.4) we define \( C^{ZZ} \) to be a partitioned matrix that is partitioned into \( p^2 \) matrices of size \( m \times m \), that is

\[
C^{ZZ} = \begin{bmatrix}
C^{ZZ}_{11} & C^{ZZ}_{12} & \cdots & C^{ZZ}_{1p} \\
C^{ZZ}_{21} & C^{ZZ}_{22} & \cdots & C^{ZZ}_{2p} \\
\vdots & \vdots & \ddots & \vdots \\
C^{ZZ}_{p1} & C^{ZZ}_{p2} & \cdots & C^{ZZ}_{pp}
\end{bmatrix} \tag{9.7}
\]

where \( C^{ZZ}_{ij} \) is an \( m \times m \) matrix corresponding to the \((i, j)\)th pair of coefficients, varieties or sires.

For simplicity we will consider the case when \( p = 3 \). Using Result A.16, it can easily be shown that when \( i = j \)

\[
G_p^{-1} G_{pij} G_p^{-1} = \begin{bmatrix}
\gamma^{11} & \gamma^{1i} & \gamma^{13} \\
\gamma^{21} & \gamma^{2i} & \gamma^{23} \\
\gamma^{31} & \gamma^{3i} & \gamma^{33}
\end{bmatrix}
\]
and hence

\[ C^{Z\!Z}(G_p^{-1} \hat{G}_{p,j} G_p^{-1} \otimes I_m) \]

Using Result A.17 and noting that \( \gamma^{ij} = \gamma^{ji} \) due to the symmetry of \( G_p^{-1} \) we find that

\[
\text{tr}(C^{Z\!Z}(G_p^{-1} \hat{G}_{p,j} G_p^{-1} \otimes I_m)) = \sum_{l=1}^{3} \sum_{k=1}^{3} \text{tr}(C^{Z\!Z}_{lk} \gamma^{ki} \gamma^{jl} I_m)
\]

and hence

\[
\text{tr}(C^{Z\!Z}(G_p^{-1} \hat{G}_{p,j} G_p^{-1} \otimes I_m)) = \sum_{l=1}^{3} \sum_{k=1}^{3} \gamma^{ki} \gamma^{jl} \text{tr}(C^{Z\!Z}_{lk})
\]

And when \( i \neq j \)

\[
G_p^{-1} \hat{G}_{p,j} G_p^{-1} = \begin{bmatrix}
\gamma^{1j} \gamma^{i1} + \gamma^{1i} \gamma^{j1} \\
\gamma^{2j} \gamma^{i2} + \gamma^{1i} \gamma^{j2} \\
\gamma^{3j} \gamma^{i3} + \gamma^{1i} \gamma^{j3}
\end{bmatrix}
\]

Using Result A.17 gives

\[
\text{tr}(C^{Z\!Z}(G_p^{-1} \hat{G}_{p,j} G_p^{-1} \otimes I_m)) = \sum_{l=1}^{3} \sum_{k=1}^{3} \text{tr}(C^{Z\!Z}_{kl} \gamma^{kj} \gamma^{il} + \gamma^{ki} \gamma^{jl} I_m)
\]

Interchanging \( k \) and \( l \) in the first sum above and noting that \( \gamma^{ij} = \gamma^{ji} \) due to the symmetry of \( G_p^{-1} \) we find

\[
\text{tr}(C^{Z\!Z}(G_p^{-1} \hat{G}_{p,j} G_p^{-1} \otimes I_m)) = 2 \sum_{l=1}^{3} \sum_{k=1}^{3} \gamma^{ik} \gamma^{jl} \text{tr}(C^{Z\!Z}_{kl})
\]
Therefore for general $p$

\[ \text{tr}(C^{ZZ}(G^{-1}_p \dot{G}_{p_j} G^{-1}_p \otimes I_m)) = \left\{ \begin{array}{ll}
\sum_{l=1}^p \sum_{k=1}^p \gamma^{lk}_{ij} \text{tr}(C^{ZZ}_{lk}), & i = j \\
2 \sum_{l=1}^p \sum_{k=1}^p \gamma^{lk}_{ij} \text{tr}(C^{ZZ}_{lk}), & i \neq j
\end{array} \right. \]

Now define $\Psi$ to be a $p \times p$ matrix whose elements are given by $\Psi_{ij} = \text{tr}(C^{ZZ}_{ij})$. Using Result A.16 and noting that $C^{ZZ}_{kl} = C^{ZZ}_{lk}$ due to the symmetry of $C^{ZZ}$, we find that

\[ G^{-1}_p \Psi G^{-1}_p = \left\{ \begin{array}{ll}
\sum_{l=1}^p \sum_{k=1}^p \gamma^{lk}_{ij} \text{tr}(C^{ZZ}_{lk})\gamma^{li}_{ji}, & i = j \\
2 \sum_{l=1}^p \sum_{k=1}^p \gamma^{lk}_{ij} \text{tr}(C^{ZZ}_{lk})\gamma^{lj}_{ij}, & i \neq j
\end{array} \right. \]

Define $\psi^*_{ij}$ to be the $(i,j)$th element of $\Psi^* = G^{-1}_p \Psi G^{-1}_p$, that is $\psi^*_{ij} = \{G^{-1}_p \Psi G^{-1}_p\}_{ij}$. Therefore

\[ \text{tr}(C^{ZZ}(G^{-1}_p \dot{G}_{p_j} G^{-1}_p \otimes I_m)) = \left\{ \begin{array}{ll}
\psi^*_{ii}, & i = j \\
2 \psi^*_{ij}, & i \neq j
\end{array} \right. \]  
(9.8)

and hence

\[ \text{tr}(C^{ZZ(m)}(G^{-1}_p \dot{G}_{p_j} G^{-1}_p \otimes I_m)) = \left\{ \begin{array}{ll}
\psi^*(m)_{ii}, & i = j \\
2 \psi^*(m)_{ij}, & i \neq j
\end{array} \right. \]

where $\psi^*(m)_{ij}$ is the $(i,j)$th element of $\Psi^*(m) = G^{-1}_p \Psi(m) G_p$.

Hence the M-step for $\gamma_{ij}$ in (9.4) can be written as

\[ \frac{\partial E_{EM}[\ell_{c}(\kappa)](m)}{\partial \gamma_{ij}} = \left\{ \begin{array}{ll}
-\frac{1}{2} \left( m\gamma_{ii} - \psi^*(m)_{ii} - \tilde{\delta}^*(m)_{ii} \right), & i = j \\
-\frac{1}{2} \left( 2m\gamma_{ij} - 2\psi^*(m)_{ij} - 2\tilde{\delta}^*(m)_{ij} \right), & i \neq j
\end{array} \right. \]

Equating to zero and solving for $\gamma^{ij}$ gives

\[ m\gamma^{ij(m+1)} = \psi^*(m)_{ij} + \tilde{\delta}^*(m)_{ij} \]

and therefore in matrix form we have

\[ mG^{(m+1)-1}_p = \Psi^*(m) + \tilde{\Delta}^*(m) \]

Pre and post multiplying by $G^{(m+1)}_p$ gives

\[ mG^{(m+1)}_p = \Psi^*(m) + \tilde{\Delta}^*(m) \]  
(9.9)

and hence

\[ m\gamma^{(m+1)}_{ij} = \psi^*(m)_{ij} + \tilde{\delta}^*(m)_{ij} \]
Recall that $\psi_{ij}^{(m)} = \text{tr}(C_{ij}^{ZZ(m)})$ and $\Delta^{(m)} = \tilde{\mathbf{u}}^{(m)'} \tilde{\mathbf{u}}^{(m)}$ where $\tilde{\Delta}$ is of the form

$$\tilde{\Delta} = \tilde{\mathbf{u}}^{(m)'} \tilde{\mathbf{u}}^{(m)} = \begin{bmatrix} u_1' u_1 & u_1' u_2 & \cdots & u_1' u_p \\ u_2' u_1 & u_2' u_2 & \cdots & u_2' u_p \\ \vdots & \vdots & \ddots & \vdots \\ u_p' u_1 & u_p' u_2 & \cdots & u_p' u_p \end{bmatrix}$$

and hence $\tilde{\delta}_{ij}^{(m)} = \tilde{u}_i^{(m)'} \tilde{u}_j^{(m)}$. Therefore the EM update for $\gamma_{ij}$ is given by

$$\gamma_{ij}^{(m+1)} = \frac{1}{m} \left( \text{tr}(C_{ij}^{ZZ(m)}) + \tilde{u}_i^{(m)'} \tilde{u}_j^{(m)} \right) \quad (9.10)$$

Recall from Chapter 4 that the EM update for $\sigma^2$ for the general linear mixed model in (2.1) is

$$\sigma^{2(m+1)} = \frac{1}{n} \left( \text{tr}(WC^{-1(m)} W'S^{-1}) + \tilde{e}^{(m)'} S^{-1} \tilde{e}^{(m)} \right)$$

For the random coefficient model, $\Sigma = I_n$ and hence the EM update for $\sigma^2$ is given by

$$\sigma^{2(m+1)} = \frac{1}{n} \left( \text{tr}(WC^{-1(m)} W') + \tilde{e}^{(m)'} \tilde{e}^{(m)} \right)$$

The rate of convergence of the REML EM algorithm is given by

$$M'_{EM}(\hat{\kappa}) = I - \mathcal{I}_{Ec}(\hat{\kappa})^{-1} \mathcal{I}_O(\hat{\kappa})$$

where

$$\mathcal{I}_{Ec}(\kappa) = E \left[ - \frac{\partial^2 \ell_c(\kappa)}{\partial \kappa \partial \kappa'} \bigg| y_2 \right]$$

The REML EM complete data log-likelihood for the unstructured $G$ model is given by

$$\ell_c(\kappa) = - \frac{1}{2} \left( m \log |G_p| + u'(G_p^{-1} \otimes I_m) u \right) - \frac{1}{2} \left( n \log \sigma^2 + \frac{e'e}{\sigma^2} \right)$$

The derivatives of $\ell_c(\kappa)$ needed to calculate $\mathcal{I}_{Ec}(\kappa)$ are

$$\frac{\partial \ell_c(\kappa)}{\partial \gamma_{ij}} = - \frac{1}{2} \left( m \text{tr}(G_p^{-1} \dot{G}_{p,i}) - u'(G_p^{-1} \dot{G}_{p,j} G_p^{-1} \otimes I_m) u \right)$$

$$\frac{\partial^2 \ell_c(\kappa)}{\partial \gamma_{ij} \partial \gamma_{kl}} = - \frac{1}{2} \left( -m \text{tr}(G_p^{-1} \dot{G}_{p,j} G_p^{-1} \dot{G}_{p,k} G_p^{-1} \otimes I_m) u \right) + 2u'(G_p^{-1} \dot{G}_{p,j} G_p^{-1} \dot{G}_{p,k} G_p^{-1} \otimes I_m) u$$

$$\frac{\partial^2 \ell_c(\kappa)}{\partial \gamma_{ij} \partial \sigma^2} = \frac{\partial^2 \ell_c(\kappa)}{\partial \sigma^2 \partial \gamma_{ij}} = 0$$

$$\frac{\partial \ell_c(\kappa)}{\partial \sigma^2} = - \frac{1}{2} \left( \frac{n}{\sigma^2} - \frac{e'e}{\sigma^4} \right)$$

$$\frac{\partial^2 \ell_c(\kappa)}{\partial (\sigma^2)^2} = - \frac{1}{2} \left( - \frac{n}{\sigma^4} + \frac{2e'e}{\sigma^6} \right)$$
Hence the elements of the REML EM conditional expected information matrix for the
complete data are

\[ I_{Ec}(\gamma_{ij}, \gamma_{kl}) = E \left[ -\frac{\partial^2 \ell_c(\kappa)}{\partial \gamma_{ij} \partial \gamma_{kl}}|y_2 \right] \]

\[ = \frac{1}{2} \left( -\text{mtr}(G_p^{-1} \dot{G}_{p_j} G_p^{-1} \dot{G}_{p_kl}) + 2E[\dot{u}(G_p^{-1} \dot{G}_{p_j} G_p^{-1} \dot{G}_{p_kl} G_p^{-1} \otimes I_m)\dot{u}|y_2] \right) \]

\[ = \frac{1}{2} \left( -\text{mtr}(G_p^{-1} \dot{G}_{p_j} G_p^{-1} \dot{G}_{p_kl}) + 2\dot{u}(G_p^{-1} \dot{G}_{p_j} G_p^{-1} \dot{G}_{p_kl} G_p^{-1} \otimes I_m)\dot{u} \]

\[ + 2\text{tr}((G_p^{-1} \dot{G}_{p_j} G_p^{-1} \dot{G}_{p_kl} G_p^{-1} \otimes I_m)C^{ZZ}) \]

\[ I_{Ec}(\gamma_{ij}, \sigma^2) = I_{Ec}(\sigma^2, \gamma_{ij}) \]

\[ = E \left[ -\frac{\partial^2 \ell_c(\kappa)}{\partial \gamma_{ij} \partial \sigma^2} | y_2 \right] \]

\[ = 0 \]

\[ I_{Ec}(\sigma^2, \sigma^2) = E \left[ -\frac{\partial^2 \ell_c(\kappa)}{\partial \sigma^2 \partial \sigma^2} | y_2 \right] \]

\[ = \frac{1}{2} \left( -\frac{n}{\sigma^4} + \frac{2E[e'|y_2]}{\sigma^6} \right) \]

\[ = \frac{1}{2} \left( -\frac{n}{\sigma^4} + 2(\dot{e}' \dot{e} + \text{tr}(WC^{-1}W')) \right) \]

Evaluating the elements of \( I_{Ec}(\kappa) \) at \( \kappa = \hat{\kappa} \) gives

\[ I_{Ec}(\hat{\gamma}_{ij}, \hat{\gamma}_{kl}) = \frac{1}{2} \left( -\text{mtr}(\hat{G}_p^{-1} \hat{G}_{p_j} \hat{G}_p^{-1} \hat{G}_{p_kl}) + 2\hat{u}'(\hat{G}_p^{-1} \hat{G}_{p_j} \hat{G}_p^{-1} \hat{G}_{p_kl} \hat{G}_p^{-1} \otimes I_m)\hat{u} \]

\[ + 2\text{tr}((\hat{G}_p^{-1} \hat{G}_{p_j} \hat{G}_p^{-1} \hat{G}_{p_kl} \hat{G}_p^{-1} \otimes I_m)C^{ZZ}) \]

\[ I_{Ec}(\hat{\gamma}_{ij}, \hat{\sigma}^2) = I_{Ec}(\hat{\sigma}^2, \hat{\gamma}_{ij}) = 0 \]

\[ I_{Ec}(\hat{\sigma}^2, \hat{\sigma}^2) = \frac{1}{2} \left( -\frac{n}{\hat{\sigma}^4} + \frac{2(\text{tr}(WC^{-1}W') + \dot{e}' \dot{e})}{\hat{\sigma}^6} \right) \]

\[ = \frac{n}{2\hat{\sigma}^4} \quad \text{(since } n\hat{\sigma}^2 = \dot{e}' \dot{e} + \text{tr}(WC^{-1}W')) \]

### 9.2.3 Computing EM updates

The EM updates given in the previous section can be computed from the REML score equations. Recall from (9.2) that the REML score for \( \gamma_{ij} \) for this model is given by

\[ U(\gamma_{ij}) = -\frac{1}{2} \left( \text{tr}(G_p^{-1} \dot{G}_{p_j} \otimes I_m) - \text{tr}((G_p^{-1} \dot{G}_{p_j} G_p^{-1} \otimes I_m)C^{ZZ}) - \dot{u}'(G_p^{-1} \dot{G}_{p_j} G_p^{-1} \otimes I_m)\dot{u} \right) \]

Using Result A.13 we have

\[ \text{tr}(G_p \dot{G}_{p_j} \otimes I_m) = \text{tr}(G_p^{-1} \dot{G}_{p_j}) \text{tr}(I_m) \]

\[ = \text{mtr}(G_p^{-1} \dot{G}_{p_j}) \]
and using Result A.13 we have

\[
\tilde{u}'(G_p^{-1}G_{ij}G_p^{-1} \otimes I_m)\tilde{u} = \text{vec}(\tilde{u})'(G_p^{-1}G_{ij}G_p^{-1} \otimes I_m)\text{vec}(\tilde{u})
\]

\[
= \text{tr}(G_p^{-1}G_{ij}G_p^{-1}\tilde{u}'I_m\tilde{u})
\]

\[
= \text{tr}(G_p^{-1}G_{ij}G_p^{-1}\tilde{u}'\tilde{u})
\]

\[
= \text{tr}(G_p^{-1}G_{ij}G_p^{-1}\tilde{\Delta})
\]

where \(\tilde{\Delta} = \tilde{u}'\tilde{u} \). We can now write the REML score for \(\gamma_{ij} \) as

\[
U(\gamma_{ij}) = -\frac{1}{2} \left( \text{mtr}(G_p^{-1}G_{ij}) - \text{tr}((G_p^{-1}G_{ij}G_p^{-1} \otimes I_m)G_{ZZ}^T) - \text{tr}(G_p^{-1}G_{ij}G_p^{-1}\tilde{\Delta}) \right)
\]

Using the results in (9.5), (9.8) and (9.6) we can write this as

\[
U(\gamma_{ij}) = \begin{cases}
-\frac{1}{2} \left( m\gamma_{ii} - \psi_{ii}^* - \delta_{ii}^* \right), & i = j \\
-\frac{1}{2} \left( 2m\gamma_{ij} - 2\psi_{ij}^* - 2\delta_{ij}^* \right), & i \neq j
\end{cases}
\]

Define \( S \) to be a \( p \times p \) upper triangular matrix, \( S = \{s_{ij}\} \) \( i \leq j \) where \( s_{ij} = U(\gamma_{ij}) \) and define \( T = \{t_{ij}\} \) to be a \( p \times p \) matrix, \( T = S + S' \) so that

\[
t_{ij} = \begin{cases}
2s_{ii}, & i = j \\
2s_{ij}, & i \neq j
\end{cases}
\]

where

\[
s_{ij} = \begin{cases}
-\frac{1}{2} \left( m\gamma_{ii} - \psi_{ii}^* - \delta_{ii}^* \right), & i = j \\
-\frac{1}{2} \left( 2m\gamma_{ij} - 2\psi_{ij}^* - 2\delta_{ij}^* \right), & i \neq j
\end{cases}
\]

Therefore

\[
t_{ij} = \psi_{ij}^* + \delta_{ij}^* - m\gamma_{ij}
\]

We can write this in matrix form

\[
T = \Psi + \tilde{\Delta} - mG_p^{-1}
\]

\[
= G_p^{-1}\Psi G_p^{-1} + G_p^{-1}\tilde{\Delta}G_p^{-1} - mG_p^{-1}
\]

Pre and post multiplying by \( G_p \) gives

\[
G_pTG_p = \Psi + \tilde{\Delta} - mG_p
\]

Rearranging and evaluating when \( \kappa = \kappa^{(m)} \) we get

\[
\Psi^{(m)} + \tilde{\Delta}^{(m)} = G_p^{(m)}T^{(m)}G_p^{(m)} + mG_p^{(m)}
\]

Recall from (9.9) that \( mG_p^{(m+1)} = \Psi^{(m)} + \tilde{\Delta}^{(m)} \). Therefore

\[
mG_p^{(m+1)} = G_p^{(m)}T^{(m)}G_p^{(m)} + mG_p^{(m)}
\]

Hence EM updates for \( \gamma \) are given by

\[
G_p^{(m+1)} = G_p^{(m)} + G_p^{(m)}T^{(m)}G_p^{(m)}/m
\]

Note that this is of a similar form to the AI updates and is easily calculated from the REML score equations.
9.2.4 PXEM Algorithm

We expand the unstructured $G$ model in (9.1) to

$$y = X\tau + Z\Lambda f + e$$

such that $u = \Lambda f$ where $f$ is a vector of rescaled random effects and $\Lambda$ is a $mp \times mp$ invertible matrix of the form $\Lambda = \Lambda_p \otimes I_m$ where $\Lambda_p$ is a $p \times p$ unstructured matrix given by

$$\Lambda_p = \begin{bmatrix}
\lambda_{11} & \lambda_{12} & \cdots & \lambda_{1p} \\
\lambda_{21} & \lambda_{22} & \cdots & \lambda_{2p} \\
\vdots & \vdots & \ddots & \vdots \\
\lambda_{p1} & \lambda_{p2} & \cdots & \lambda_{pp}
\end{bmatrix}$$

and $\lambda = \text{vec}(\Lambda_p) = (\lambda_{11}, \lambda_{21}, \ldots, \lambda_{p1}, \lambda_{12}, \lambda_{22}, \ldots, \lambda_{p2}, \ldots, \lambda_{1p}, \lambda_{2p}, \ldots, \lambda_{pp})'$.

The joint distribution of $f$ and $e$ is given by

$$\begin{bmatrix} f \\ e \end{bmatrix} \sim N\left( \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} D & 0 \\ 0 & \sigma^2 I_n \end{bmatrix} \right)$$

where $D$ is of the same form as $G$, that is $D = (D_p \otimes I_m)$ where $D_p$ is a symmetric unstructured variance matrix. $D_p$ is a function of $d$ where $d = \text{vech}(D_p) = (d_{11}, d_{12}, \ldots, d_{1p}, d_{22}, \ldots, d_{2p}, \ldots, d_{pp})'$.

Note that

$$\text{Var}(u) = \Lambda \text{Var}(f) \Lambda$$

$$G = \Lambda D \Lambda'$$

$$(G_p \otimes I_m) = (\Lambda_p \otimes I_m)(D_p \otimes I_m)(\Lambda_p' \otimes I_m)$$

$$G_p = \Lambda_p D_p \Lambda_p'$$

The variance parameters of the expanded model are $K' = (\kappa', \lambda')$ where $\kappa' = (d', \sigma^2)$. The null value of $\lambda$ is $\lambda_0 = \text{vec}(I_p)$ and hence $\Lambda_p = I_p$ and $\Lambda_0 = I_p \otimes I_m = I_{mp}$. Note that when $\lambda$ is set to its null value $\lambda_0$, the expanded model is reduced to the unstructured $G$ model given in (9.1).

The PXEM update for $d_{ij}$ is equivalent to the EM update for $\gamma_{ij}$ and is given by

$$d_{ij}^{(m+1)} = \frac{1}{m} \left( \text{tr}(G_{ij}^{ZZ(m)}) + \hat{u}_{i}^{(m)}' \hat{u}_{j}^{(m)} \right)$$

The PXEM update for $\sigma^2$ is the equivalent to the EM update for $\sigma^2$ and is given by

$$\sigma^{2(m+1)} = \frac{1}{n} \left( \text{tr}(WC^{-1(m)}W') + \hat{e}^{(m)}' \hat{e}^{(m)} \right)$$

Noting that for the unstructured $G$ model the PXEM update for $\lambda$ is given by

$$\lambda^{(m+1)} = A^{(m)} b^{(m)}$$
where $A^{(m)} = \{a_{ij,kl}^{(m)}\}$ and $b^{(m)} = \{b_{ij}^{(m)}\}$ such that

$$a_{ij,kl}^{(m)} = \tilde{u}_{ij}^{(m)} Z' Z \hat{\lambda}_{kl}^{(m)} + \text{tr}(\hat{\lambda}_{ij}^{(m)} Z' Z \hat{\lambda}_{kl}^{(m)} C^{ZZ(m)})$$

and

$$b_{ij}^{(m)} = \tilde{u}_{ij}^{(m)} Z' (y - X \hat{\tau}^{(m)}) - \text{tr}(\hat{\lambda}_{ij}^{(m)} Z' X C^{XZ(m)})$$

Note that

$$\hat{\lambda}_{ij} = \frac{\partial \Lambda}{\partial \lambda_{ij}} = \frac{\partial (\Lambda_p \otimes I_m)}{\partial \lambda_{ij}} = \frac{\partial \Lambda_p}{\partial \lambda_{ij}} \otimes I_m = \hat{\lambda}_{p_{ij}} \otimes I_m$$

where $\hat{\lambda}_{p_{ij}} = \frac{\partial A_p}{\partial \lambda_{ij}}$. Recall that the vector of random effects is composed of $p$ subvectors of length $m \times 1$ such that $u' = (u_1', u_2', \ldots, u_p')$ where $u_i' = (u_{i1}, u_{i2}, \ldots, u_{mi})$ is the vector of random coefficients for the $i$th subject, random effect for the $i$th variety or sire. Each subvector has an associated design matrix $Z_i$ of size $n \times m$ such that $Z$ is of the form $Z = [Z_1 \ Z_2 \ \cdots \ Z_p]$. It can easily be shown that

$$Z \hat{\lambda}_{ij} u = Z_i u_j$$

and hence

$$\tilde{u}_{ij}^{(m)} Z' Z \hat{\lambda}_{ij} u^{(m)} = \tilde{u}_{ij}^{(m)} Z_i Z_k \tilde{u}_l^{(m)}$$

and

$$\tilde{u}_{ij}^{(m)} Z' (y - X \hat{\tau}^{(m)}) = \tilde{u}_{ij}^{(m)} Z_i (y - X \hat{\tau}^{(m)})$$

Recall that $C^{ZZ}$ can be partitioned as shown in (9.7). It can easily be shown that

$$\text{tr}(\hat{\lambda}_{ij}^{(m)} Z' Z \hat{\lambda}_{kl}^{(m)} C^{ZZ}) = \text{tr}(Z_i' Z_k C^{ZZ}_{jl})$$

Note that $C^{XZ}$ can also be partitioned into $p$ matrices of size $t \times m$, that is $C^{XZ} = \left[\begin{array}{ccc} C_1^{XZ} & \cdots & C_p^{XZ} \end{array}\right]$. It can then be shown that

$$\text{tr}(\hat{\lambda}_{ij}^{(m)} Z' X C^{XZ(m)}) = \text{tr}(Z_i' X C_j^{XZ(m)})$$

Hence the PXEM update for $\lambda$ is given by

$$\lambda^{(m+1)} = A^{(m)}^{-1} b^{(m)}$$

where $A^{(m)} = \{a_{ij,kl}^{(m)}\}$ and $b^{(m)} = \{b_{ij}^{(m)}\}$ such that

$$a_{ij,kl}^{(m)} = \tilde{u}_{ij}^{(m)} Z' Z \tilde{u}_l^{(m)} + \text{tr}(Z_i' Z_k C^{ZZ}_{jl}^{(m)})$$

and

$$b_{ij}^{(m)} = \tilde{u}_{ij}^{(m)} Z_i (y - X \hat{\tau}^{(m)}) - \text{tr}(Z_i' X C_j^{XZ(m)})$$
9.2. UNSTRUCTURED G MODEL AND RESULTS

To obtain the update for \( \mathbf{G}_p(\gamma) \) we form \( \mathbf{D}_p^{(m+1)} \) and \( \Lambda_p^{(m+1)} \) and calculate

\[
\mathbf{G}_p^{(m+1)} = \Lambda_p^{(m+1)} \mathbf{D}_p^{(m+1)} \Lambda_p^{(m+1)'}
\]

The convergence rate matrix for the PXEM algorithm is given by

\[
\mathbf{M'}_{pX}(\hat{\mathbf{K}}) = \mathbf{I} - \mathbf{I}_{Ec}(\hat{\mathbf{K}})^{-1} \mathbf{I}_O(\hat{\mathbf{K}})
\]

where

\[
\mathbf{I}_{Ec}(\mathbf{K}) = \mathbb{E} \left[ -\frac{\partial^2 \ell_c(\mathbf{K})}{\partial \mathbf{K} \partial \mathbf{K}'} \mid \mathbf{y}_2 \right]
\]

Note that \( \mathbf{K} = (\mathbf{\kappa}, \mathbf{\lambda}) = (\gamma, \sigma^2, \lambda) \).

The REML PXEM complete data log-likelihood for the unstructured \( \mathbf{G} \) model is given by

\[
\ell_c(\mathbf{K}) = -\frac{1}{2} \left( m \log |\mathbf{D}_p| + f'(\mathbf{D}_p^{-1} \otimes \mathbf{I}_m) \mathbf{f} \right) - \frac{1}{2} \left( n \log \sigma^2 + \frac{\mathbf{e}'\mathbf{e}}{\sigma^2} \right)
\]

\[
= -\frac{1}{2} \left( m \log |\Lambda_p^{-1} \mathbf{G}_p \Lambda_p^{-1}'| + f'(\Lambda_p' \mathbf{G}_p^{-1} \Lambda_p' \otimes \mathbf{I}_m) \mathbf{f} \right) - \frac{1}{2} \left( n \log \sigma^2 + \frac{\mathbf{e}'\mathbf{e}}{\sigma^2} \right)
\]

since \( \mathbf{D}_p = \Lambda_p^{-1} \mathbf{G}_p \Lambda_p^{-1}' \).

Note that \( \mathbf{e} = \mathbf{y} - \mathbf{X}\tau - \mathbf{Z}\lambda \mathbf{f} \) and hence

\[
\mathbf{e}'\mathbf{e} = (\mathbf{y} - \mathbf{X}\tau)'(\mathbf{y} - \mathbf{X}\tau) - 2\mathbf{f}'\mathbf{Z}'\Lambda'(\mathbf{y} - \mathbf{X}\tau) + \mathbf{f}'\mathbf{Z}'\Lambda\Lambda\mathbf{Z}\mathbf{f}
\]

and

\[
\frac{\partial \mathbf{e}'\mathbf{e}}{\partial \lambda_{ij}} = -2\mathbf{f}'\hat{\Lambda}_{ij}'\mathbf{Z}'(\mathbf{y} - \mathbf{X}\tau) + 2\mathbf{f}'\hat{\Lambda}_{ij}'\mathbf{Z}'\Lambda\mathbf{f}
\]

The derivatives of \( \ell_c(\mathbf{K}) \) needed to calculate \( \mathbf{I}_{Ec}(\mathbf{K}) \) are

\[
\frac{\partial \ell_c(\mathbf{K})}{\partial \gamma_{ij}} = -\frac{1}{2} \left( \text{tr}((\mathbf{G}_p^{-1} \mathbf{G}_p^{-1}) - f'(\Lambda_p' \mathbf{G}_p^{-1} \mathbf{G}_p^{-1} \Lambda_p' \otimes \mathbf{I}_m) \mathbf{f}) \right)
\]

\[
\frac{\partial^2 \ell_c(\mathbf{K})}{\partial \gamma_{ij} \partial \gamma_{kl}} = -\frac{1}{2} \left( -\text{tr}((\mathbf{G}_p^{-1} \mathbf{G}_p^{-1} \mathbf{G}_p^{-1} \otimes \mathbf{I}_m) \mathbf{f}) \right)
\]

\[
\frac{\partial^2 \ell_c(\mathbf{K})}{\partial \gamma_{ij} \partial \sigma^2} = \frac{\partial^2 \ell_c(\mathbf{K})}{\partial \sigma^2 \partial \gamma_{ij}} = 0
\]

\[
\frac{\partial^2 \ell_c(\mathbf{K})}{\partial \lambda_{ij} \partial \lambda_{kl}} = \frac{\partial^2 \ell_c(\mathbf{K})}{\partial \lambda_{kl} \partial \gamma_{ij}} = -\frac{1}{2} \left( -2f'(\Lambda_p' \mathbf{G}_p^{-1} \mathbf{G}_p^{-1} \mathbf{G}_p^{-1} \Lambda_p' \otimes \mathbf{I}_m) \mathbf{f} \right)
\]

\[
\frac{\partial \ell_c(\mathbf{K})}{\partial \sigma^2} = -\frac{1}{2} \left( \frac{n}{\sigma^2} - \frac{\mathbf{e}'\mathbf{e}}{\sigma^4} \right)
\]

\[
\frac{\partial^2 \ell_c(\mathbf{K})}{\partial (\sigma^2)^2} = -\frac{1}{2} \left( \frac{n}{\sigma^4} + \frac{2\mathbf{e}'\mathbf{e}}{\sigma^6} \right)
\]
\[
\frac{\partial^2 \ell_c(\mathbf{K})}{\partial \sigma^2 \partial \lambda_{kl}} = \frac{\partial^2 \ell_c(\mathbf{K})}{\partial \lambda_{kl} \partial \sigma^2} = -\frac{1}{2} \left( \frac{2f'\Lambda'_{kl}Z'(y - X \tau) - 2f'\Lambda'_{kl}Z'Z\Lambda f}{\sigma^4} \right)
\]

\[
\frac{\partial \ell_c(\mathbf{K})}{\partial \lambda_{ij}} = -\frac{1}{2} \left( -2mtr(\Lambda_{p}^{-1}\Lambda_{p}^{-1}) + 2f'(\Lambda_{p}^{-1}\Lambda_{p}^{-1}\Lambda_p \otimes I_m) f \right)
\]

\[
\frac{\partial^2 \ell_c(\mathbf{K})}{\partial \lambda_{ij} \partial \lambda_{kl}} = -\frac{1}{2} \left( 2mtr(\Lambda_{p}^{-1}\Lambda_{p}^{-1}\Lambda_{p}^{-1}) + 2f'(\Lambda_{p}^{-1}\Lambda_{p}^{-1}\Lambda_p \otimes I_m) f \right)
\]

Hence the elements of the REML PXEM conditional expected information matrix for the complete data are

\[
I_{Ec}(\gamma_{ij}, \gamma_{kl}) = \mathbb{E} \left[ \frac{-\partial^2 \ell_c(\mathbf{K})}{\partial \gamma_{ij} \partial \gamma_{kl}} \right] \mathbf{y}_2
\]

\[
= \frac{1}{2} \left( -mtr(\mathbf{G}_p^{-1}\mathbf{G}_{pji}\mathbf{G}_p^{-1}\mathbf{G}_{pkl}) + 2\mathbb{E} \left[ f'(\mathbf{A}_p^{'}\mathbf{G}_p^{-1}\mathbf{G}_{pji}\mathbf{G}_p^{-1}\mathbf{G}_{pkl}\mathbf{A}_p \otimes I_m) f | \mathbf{y}_2 \right] \right)
\]

\[
= \frac{1}{2} \left( -mtr(\mathbf{G}_p^{-1}\mathbf{G}_{pji}\mathbf{G}_p^{-1}\mathbf{G}_{pkl}) + 2\mathbb{E} \left[ f'(\mathbf{A}_p^{'}\mathbf{G}_p^{-1}\mathbf{G}_{pji}\mathbf{G}_p^{-1}\mathbf{G}_{pkl}\mathbf{A}_p \otimes I_m) f | \mathbf{y}_2 \right] \right)
\]

\[
= \frac{1}{2} \left( -mtr(\mathbf{G}_p^{-1}\mathbf{G}_{pji}\mathbf{G}_p^{-1}\mathbf{G}_{pkl}\mathbf{A}_p \otimes I_m) \mathbf{f} + 2\mathbb{E} \left[ f'(\mathbf{A}_p^{'}\mathbf{G}_p^{-1}\mathbf{G}_{pji}\mathbf{G}_p^{-1}\mathbf{G}_{pkl}\mathbf{A}_p \otimes I_m) f | \mathbf{y}_2 \right] \right)
\]

\[
I_{Ec}(\gamma_{ij}, \sigma^2) = I_{Ec}(\sigma^2, \gamma_{ij})
\]

\[
= \mathbb{E} \left[ \frac{-\partial^2 \ell_c(\mathbf{K})}{\partial \gamma_{ij} \partial \sigma^2} \right] \mathbf{y}_2
\]

\[
= 0
\]

\[
I_{Ec}(\gamma_{ij}, \lambda_{kl}) = I_{Ec}(\lambda_{kl}, \gamma_{ij})
\]

\[
= \mathbb{E} \left[ \frac{-\partial^2 \ell_c(\mathbf{K})}{\partial \gamma_{ij} \partial \lambda_{kl}} \right] \mathbf{y}_2
\]

\[
= \frac{1}{2} \left( -2\mathbb{E} \left[ f'(\mathbf{A}_p^{'}\mathbf{G}_p^{-1}\mathbf{G}_{pji}\mathbf{G}_p^{-1}\mathbf{A}_p \otimes I_m) f | \mathbf{y}_2 \right] \right)
\]

\[
= \frac{1}{2} \left( -2\mathbb{E} \left[ f'(\mathbf{A}_p^{'}\mathbf{G}_p^{-1}\mathbf{G}_{pji}\mathbf{G}_p^{-1}\mathbf{A}_p \otimes I_m) f | \mathbf{y}_2 \right] \right)
\]

\[
I_{Ec}(\sigma^2, \sigma^2)
\]

\[
= \mathbb{E} \left[ \frac{-\partial^2 \ell_c(\mathbf{K})}{\partial (\sigma^2)^2} \right] \mathbf{y}_2
\]

\[
= \frac{1}{2} \left( -\frac{n}{\sigma^4} + 2\mathbb{E}[\mathbf{e}'\mathbf{e}| \mathbf{y}_2] \right)
\]

\[
= \frac{1}{2} \left( -\frac{n}{\sigma^4} + 2\mathbb{E}[\mathbf{e}'\mathbf{e}| \mathbf{y}_2] \right)
\]
\[ I_E(c, \lambda_{kl}) = I_E(\lambda_{kl}, c^2) \]
\[ = E \left[ -\frac{\partial^2 \ell_e(K)}{\partial c^2 \partial \lambda_{kl}} \right] y_2 \]
\[ = \frac{1}{2} \left( \frac{2E \left[ f' \hat{\Lambda}'_{kl} Z'(y - X) \right] y_2}{\sigma^4} - 2E \left[ f' \hat{\Lambda}'_{kl} Z' Z \Lambda f \right] y_2 \right) \]
\[ = \frac{1}{2} \left( \frac{2\hat{f}' \hat{\Lambda}'_{kl} Z'(y - X \hat{\tau}) - 2tr(\hat{\Lambda}'_{kl} Z' X C_p^{XZ}) - 2\hat{f}' \hat{\Lambda}'_{kl} Z' Z \hat{\Lambda} \hat{f} - 2tr(\hat{\Lambda}'_{kl} Z' Z \Lambda C_p^{ZZ})}{\sigma^4} \right) \]

\[ I_E(\lambda_{ij}, \lambda_{kl}) \]
\[ = E \left[ -\frac{\partial^2 \ell_e(K)}{\partial \lambda_{ij} \partial \lambda_{kl}} \right] y_2 \]
\[ = \frac{1}{2} \left( 2mtr(\hat{\Lambda}_{p_i j} \Lambda_p^{-1} \hat{\Lambda}_{p_k l} \Lambda_p^{-1}) + 2E \left[ f' (\hat{\Lambda}_{p_i j} G_p^{-1} \hat{\Lambda}_{p_k l} \otimes I_m) f \right] y_2 \right) \]
\[ + \frac{1}{2} \left( \frac{2E \left[ f' \hat{\Lambda}'_{ij} Z' Z \Lambda_{kl} f \right] y_2}{\sigma^2} \right) \]
\[ = \frac{1}{2} \left( 2mtr(\hat{\Lambda}_{p_i j} \Lambda_p^{-1} \hat{\Lambda}_{p_k l} \Lambda_p^{-1}) + 2\hat{f}' (\hat{\Lambda}_{p_i j} G_p^{-1} \hat{\Lambda}_{p_k l} \otimes I_m) \hat{f} + 2tr((\hat{\Lambda}_{p_i j} G_p^{-1} \hat{\Lambda}_{p_k l} \otimes I_m) C_p^{ZZ}) \right) \]
\[ + \frac{1}{2} \left( \frac{2\hat{f}' \hat{\Lambda}'_{ij} Z' Z \hat{\Lambda}_{kl} \hat{f} + 2tr(\hat{\Lambda}'_{ij} Z' Z \Lambda_{kl} C_p^{ZZ})}{\sigma^2} \right) \]

Evaluating the elements of \( I_E(K) \) at \( K = (\hat{K}, \lambda_0) \), where \( \Lambda_{p_0} = I_p \) and \( \Lambda_0 = I_{mp} \), gives

\[ I_E(\hat{\gamma}_{ij}, \hat{\gamma}_{kl}) \]
\[ = \frac{1}{2} \left( -mtr(G_p^{-1} \hat{G}_{p_j i} G_p^{-1} \hat{G}_{p_l k} \otimes I_m) + 2\hat{u}' (\hat{G}_p^{-1} \hat{G}_{p_j i} \hat{G}_p^{-1} \hat{G}_{p_l k} \otimes I_m) u \right) \]
\[ + 2tr((\hat{G}_p^{-1} \hat{G}_{p_j i} \hat{G}_p^{-1} \hat{G}_{p_l k} \otimes I_m) C_p^{ZZ}) \]
\[ I_E(\hat{\gamma}_{ij}, \hat{\sigma}^2) = I_E(\hat{\sigma}^2, \hat{\gamma}_{ij}) \]
\[ = 0 \]
\[ I_E(\hat{\gamma}_{ij}, \lambda_{0_k l}) = I_E(\lambda_{0_k l}, \hat{\gamma}_{ij}) \]
\[ = \frac{1}{2} \left( -2\hat{u}' (\hat{G}_p^{-1} \hat{G}_{p_j i} \hat{G}_p^{-1} \hat{L}_{p_l k} \otimes I_m) \hat{u} - 2tr((\hat{G}_p^{-1} \hat{G}_{p_j i} \hat{G}_p^{-1} \hat{L}_{p_l k} \otimes I_m) C_p^{ZZ}) \right) \]
\[ I_E(\hat{\sigma}^2, \hat{\sigma}^2) \]
\[ = \frac{1}{2} \left( -n \frac{\hat{c}^2}{\sigma^4} + 2 \left( tr(WC^{-1}W') + \hat{c}' \hat{\sigma} \right) \right) \]
\[ = \frac{n}{2\sigma^4} \quad \text{(since } n\hat{c}^2 = \hat{c}' \hat{\sigma} + tr(WC^{-1}W')) \]
\[ I_E(\hat{\sigma}^2, \lambda_{0_k l}) = I_E(\lambda_{0_k l}, \hat{\sigma}^2) \]
\[ = 0 \quad \text{(see below)} \]
\[ I_E(\lambda_{0_k l}, \lambda_{0_k l}) \]
\[= \frac{1}{2} \left( 2\text{tr}(\Lambda_{p_i} \Lambda_{p_k}) + 2\tilde{u}'(\Lambda'_{p_i} G^{-1}_p \Lambda_{p_k} \otimes I_m) \tilde{u} + 2\text{tr}\left((\Lambda'_{p_i} G^{-1}_p \Lambda_{p_k} \otimes I_m)C^{ZZ}\right)\right) + \frac{1}{2} \left( \frac{2\tilde{u}' \Lambda'_{ij} Z' Z \Lambda_{kl} \tilde{u} + 2\text{tr}\left(\Lambda'_{ij} Z' Z \Lambda_{kl} C^{ZZ}\right)}{\hat{\sigma}^2} \right) \]

To show that \( I_{Ec}(\hat{\sigma}^2, \lambda_{0kl}) = 0 \) consider \( I_{Ec}(\sigma^2, \lambda_{kl}) \) at convergence, that is evaluated at \( \kappa = \hat{\kappa} \) and \( \lambda = \hat{\lambda} \).

\[ I_{Ec}(\hat{\sigma}^2, \hat{\lambda}_{kl}) = \frac{\tilde{u}' \hat{\Lambda}'_{kl} Z'(y - X \hat{\tau}) - \text{tr}(\hat{\Lambda}'_{kl} Z' X C^{XZ}) - \tilde{u}' \hat{\Lambda}'_{kl} Z' Z \hat{\Lambda} \tilde{u} - \text{tr}(\hat{\Lambda}'_{kl} Z' Z \hat{\Lambda} C^{ZZ})}{\hat{\sigma}^4} \]

Recall from (5.23) that \( \Lambda = \sum_{i=1}^{mp} \sum_{j=1}^{mp} \lambda_{ij} \hat{\Lambda}_{ij} \) and hence

\[
\begin{align*}
\tilde{u}' \hat{\Lambda}'_{kl} Z' Z \hat{\Lambda} \tilde{u} + \text{tr}(\hat{\Lambda}'_{kl} Z' Z \hat{\Lambda} C^{ZZ}) \\
= \sum_{i=1}^{mp} \sum_{j=1}^{mp} \hat{\lambda}_{ij} \tilde{u}' \hat{\Lambda}'_{kl} Z' Z \hat{\Lambda}_{ij} \tilde{u} + \text{tr}\left(\sum_{i=1}^{mp} \sum_{j=1}^{mp} \hat{\lambda}_{ij} \hat{\Lambda}'_{kl} Z' Z \hat{\Lambda}_{ij} C^{ZZ}\right) \\
= \sum_{i=1}^{mp} \sum_{j=1}^{mp} \hat{\lambda}_{ij} \left( \tilde{u}' \hat{\Lambda}'_{kl} Z' Z \hat{\Lambda}_{ij} \tilde{u} + \text{tr}(\hat{\Lambda}'_{kl} Z' Z \hat{\Lambda}_{ij} C^{ZZ}) \right) \\
= \sum_{i=1}^{mp} \sum_{j=1}^{mp} \hat{\lambda}_{ij} a_{kl,ij} \\
= b_{kl} \\
= \tilde{u}' \hat{\Lambda}'_{kl} Z'(y - X \hat{\tau}) - \text{tr}(\hat{\Lambda}'_{kl} Z' X C^{XZ})
\end{align*}
\]

Therefore \( I_{Ec}(\hat{\sigma}^2, \hat{\lambda}_{kl}) = 0 \) and hence \( I_{Ec}(\hat{\sigma}^2, \lambda_{0kl}) = 0 \).

Recall that the convergence of the PXEM algorithm is determined by the convergence of \( \kappa^{(m)} \), and the rate matrix for \( \kappa \) for the PXEM algorithm is given by

\[ M'_{PX}(\hat{\kappa}) = I - V_{\hat{\kappa} \hat{\kappa}} I_{Ec}(\hat{\kappa}) \]

where

\[ I_{Ec}(\hat{\kappa})^{-1} = \begin{bmatrix} V_{\hat{\kappa} \hat{\kappa}} & V_{\hat{\kappa} \lambda_0} \\ V_{\lambda_0 \hat{\kappa}} & V_{\lambda_0 \lambda_0} \end{bmatrix} \]
9.2.5 Local EM Scheme

The local data is defined as \( \mathbf{v} = \Omega^{s(m)}(m-1) Z S^{(m)} y \) where 
\[
\Omega^{s(m)} = (\mathbf{C} \mathbf{Z} \mathbf{Z}^T)^{-1} - \mathbf{G}^{(m)}^{-1}
\]
and \( \mathbf{C} \mathbf{Z} \mathbf{Z}^T \) is an approximation of \( \mathbf{C} \mathbf{Z} \mathbf{Z}^T \) such that 
\[
\mathbf{C} \mathbf{Z} \mathbf{Z}^T = \text{diag}(\mathbf{C} \mathbf{Z} \mathbf{Z}^T_{i,j}).
\]
The local data has distribution \( \mathbf{v} \sim N(\mathbf{0}, \mathbf{G} + \Omega^{s(m)}(m-1)) \) where 
\[
\mathbf{G} = \mathbf{G}_p \otimes \mathbf{I}_m.
\]
The local variance components model is given by
\[
\mathbf{v} = \mathbf{u}_v + \mathbf{e}_v \tag{9.11}
\]
where
\[
\begin{bmatrix} \mathbf{u}_v \\ \mathbf{e}_v \end{bmatrix} \sim N \left( \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \mathbf{G} & \mathbf{0} \\ \mathbf{0} & (\Omega^{s(m)}(m-1))^{-1} \end{bmatrix} \right)
\]
The BLUP \( \hat{\mathbf{u}}_v \) is given by
\[
\hat{\mathbf{u}}_v = \mathbf{C}_v^{-1} \Omega^{s(m)} \mathbf{v}
\]
where \( \mathbf{C}_v = \Omega^{s(m)} + \mathbf{G}^{-1} \) is the coefficient matrix of the mixed model equations for the local model.

Note that \( \hat{\mathbf{u}}_v \) can be partitioned into \( p \) subvectors of length \( m \times 1 \) for each subject, that is 
\( \hat{\mathbf{u}}_v = (\hat{\mathbf{u}}_{v1}, \hat{\mathbf{u}}_{v2}, \ldots, \hat{\mathbf{u}}_{vp}) \) where \( \hat{\mathbf{u}}_{vi} = (\hat{u}_{v1i}, \hat{u}_{v2i}, \ldots, \hat{u}_{vmi}) \). And \( \mathbf{C}_v^{-1} \) can be partitioned into \( p^2 \) submatrices of size \( m \times m \), that is \( \mathbf{C}_v^{-1} = \{ \mathbf{C}_v^{-1}_{ij} \} \).

The local M-step for \( \gamma_{ij} \)
\[
\frac{\partial \ell_{EM}(\mathbf{u}_v)_{(n)}}{\partial \gamma_{ij}} = -\frac{1}{2} \left( \text{tr}(\mathbf{G}^{-1} \hat{\mathbf{G}}_{ij}) - \text{tr}(\mathbf{C}_v^{-1(n)} \mathbf{G}^{-1} \hat{\mathbf{G}}_{ij} \mathbf{G}^{-1} - \hat{\mathbf{u}}_{v(n)} \mathbf{G}^{-1} \hat{\mathbf{G}}_{ij} \mathbf{G}^{-1} \hat{\mathbf{u}}_{v(n)}) \right)
\]

Following from section (9.2.2), the first term in the local M-step for \( \gamma_{ij} \) for the unstructured \( \mathbf{G} \) model is given by
\[
\text{tr}(\mathbf{G}^{-1} \hat{\mathbf{G}}_{ij}) = m \text{tr}(\mathbf{G}^{-1}_p \hat{\mathbf{G}}_{p_{ij}})
\]
\[
= \begin{cases} 
\gamma_{ij}, & i = j \\
2\gamma_{ij}, & i \neq j 
\end{cases}
\]
where \( \gamma_{ij}, i \leq j = 1, 2, \ldots, p \) are the unique elements of \( \mathbf{G}_p^{-1} \). The second term is given by
\[
\text{tr}(\mathbf{C}_v^{-1(n)} \mathbf{G}^{-1} \hat{\mathbf{G}}_{ij} \mathbf{G}^{-1}) = \text{tr}(\mathbf{C}_v^{-1(n)}(\mathbf{G}^{-1}_p \hat{\mathbf{G}}_{p_{ij}} \mathbf{G}^{-1}_p \otimes \mathbf{I}_m))
\]
\[
= \begin{cases} 
\psi_{v_{ij}}, & i = j \\
2\psi_{v_{ij}}, & i \neq j 
\end{cases}
\]
where \( \Psi_v(n) = \{ \psi_{v_{ij}} \} \) where \( \psi_{v_{ij}} = \text{tr}(\mathbf{C}^{-1}_{v_{ij}}) \), and \( \Psi_v(n) = \mathbf{G}_p^{-1} \Psi_v(n) \mathbf{G}_p^{-1} = \{ \psi_{v_{ij}} \} \). The third term is given by
\[
\hat{\mathbf{u}}_{v(n)} \mathbf{G}^{-1} \hat{\mathbf{G}}_{ij} \mathbf{G}^{-1} \hat{\mathbf{u}}_{v(n)} = \text{tr}(\mathbf{G}^{-1}_p \hat{\mathbf{G}}_{p_{ij}} \mathbf{G}^{-1}_p \hat{\mathbf{A}}_{v(n)})
\]
\[
= \begin{cases} 
\delta_{v_{ij}}, & i = j \\
2\delta_{v_{ij}}, & i \neq j 
\end{cases}
\]
where $\tilde{U}_v^{(n)}$ is the $m \times p$ matrix such that $\text{vec}(\tilde{U}_v^{(n)}) = \tilde{u}_v^{(n)}$, $\tilde{\Delta}_v^{(n)} = \tilde{U}_v^{(n)'} \tilde{U}_v^{(n)}$ and $\Delta_v^{(n)} = G_p^{-1} \tilde{\Delta}_v^{(n)} G_p^{-1} = \{\tilde{\delta}_{vij}\}$.

We can now write the local M-step for $\gamma_{ij}$ as

$$\frac{\partial E_{EM}[\ell_c(u_v)](n)}{\partial \gamma_{ij}} = \begin{cases} -\frac{1}{2} \left( m \gamma_{ii} - \psi_{vii}^{(n)} - \tilde{\delta}_{vii}^{(n)} \right), & i = j \\ -\frac{1}{2} \left( 2m \gamma_{ij} - 2\psi_{vij}^{(n)} - 2\tilde{\delta}_{vij}^{(n)} \right), & i \neq j \end{cases}$$

Equating to zero and solving for $\gamma_{ij}$ gives

$$m \gamma_{ij}^{(n+1)} = \psi_{vij}^{(n)} + \tilde{\delta}_{vij}^{(n)}$$

and therefore in matrix form we have

$$m G_p^{(n+1)-1} = \Psi_v^{(n)} + \tilde{\Delta}_v^{(n)}$$

$$= G_p^{-1} \Psi_v^{(n)} G_p^{-1} + G_p^{-1} \tilde{\Delta}_v^{(n)} G_p^{-1}$$

Pre and post multiplying by $G_p^{(n+1)}$ gives

$$m G_p^{(n+1)} = \Psi_v^{(n)} + \tilde{\Delta}_v^{(n)}$$

and hence

$$m \gamma_{ij}^{(n+1)} = \psi_{vij}^{(n)} + \tilde{\delta}_{vij}^{(n)}$$

Recall that $\psi_{vij}^{(n)} = \text{tr}(C_{vij}^{-1(n)})$, and $\tilde{\Delta}_v^{(n)} = \tilde{U}_v^{(n)'} \tilde{U}_v^{(n)}$ and hence $\tilde{\delta}_{vij} = \tilde{u}_{vi}^{(n)'} \tilde{u}_{vj}^{(n)}$. Therefore the local EM update for $\gamma_{ij}$ is given by

$$\gamma_{ij}^{(n+1)} = \frac{\text{tr}(C_{vij}^{-1(n)}) + \tilde{u}_{vi}^{(n)'} \tilde{u}_{vj}^{(n)}}{m}$$

Note that on the first iteration of the local EM iteration $C_v^{(n)} = (C_{ZZ}^{(m)})^{-1}$ and $\tilde{u}_v^{(n)} = \tilde{u}^{(m)}$, therefore the local EM update for $\gamma_{ij}$ is given by

$$\gamma_{ij}^{(n+1)} = \frac{\text{tr}(C_{ij}^{ZZ}^{(m)}) + \tilde{u}_i^{(m)'} \tilde{u}_j^{(m)}}{m}$$

which is the same as the EM update for $\gamma_{ij}$ as given in (9.10). Hence the update for $\gamma_{ij}$ from the first iteration of the local EM scheme is equivalent to the update for $\gamma_{ij}$ from a full EM iteration.
9.2.6 Local PXEM Scheme

The local model in (9.2.5) is expanded to

\[ v = \Lambda f_v + e_v \]

such that \( u_v = \Lambda f_v \) where \( f \) is a vector of rescaled random effects and \( \Lambda = \Lambda_p \otimes I_m \).

The joint distribution of \( f_v \) and \( e_v \) is given by

\[
\begin{bmatrix} f_v \\ e_v \end{bmatrix} \sim N \left( \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} D & 0 \\ 0 & \Omega^{(m)(-1)} \end{bmatrix} \right)
\]

where \( D = D_p \otimes I_m \). Note that \( G_p = \Lambda_p D_p \Lambda_p' \).

The local PXEM update for \( d_{ij} \) is equivalent to the local EM update for \( \gamma_{ij} \) and is given by

\[
d_{ij}^{(n+1)} = \frac{\text{tr}(C_{vij}^{-1}(n)) + \tilde{u}_{v_i}^{(n)} \tilde{u}_{v_j}^{(n)}}{m}
\]

Following from section (9.2.4), the local PXEM update for \( \lambda \) is given by

\[
\lambda^{(n+1)} = A_v^{(n)-1} b_v^{(n)}
\]

where \( A_v^{(n)} = \{a_v^{(n)} \}_{v_{ij,kl}} \) and \( b_v^{(n)} = \{b_v^{(n)} \}_{vi} \) such that

\[
a_v^{(n)} = \tilde{u}_{v_i}^{(n)} \Lambda_{ij}' \Omega^{(m)} \Lambda_{kl} \tilde{u}_{v_j}^{(n)} + \text{tr}(\Lambda_{ij}' \Omega^{(m)} \Lambda_{kl} C_{vij}^{-1}(n))
\]

and

\[
b_v^{(n)} = \tilde{u}_{v_i}^{(n)} \Lambda_{ij}' \Omega^{(m)} \Lambda_{ij} \tilde{u}_{v_j}^{(n)} + \text{tr}(\Lambda_{ij} \Omega^{(m)} C_{vij}^{-1}(n))
\]

To obtain the local PXEM update for \( G_p(\gamma) \) we form \( D_p^{(n+1)} \) and \( \Lambda_p^{(n+1)} \) and calculate

\[
G_p^{(n+1)} = \Lambda_p^{(n+1)} D_p^{(n+1)} \Lambda_p^{(n+1)'}
\]
9.2.7 Starting Values

9.2.7.1 Multivariate Data

First we will develop a method that can be used to obtain starting values for the variance parameters for data such as multi-environment plant variety data and multi-trait animal breeding data. This method will be developed in the context of the analysis of multi-environment plant variety data. Consider a balanced multi-environment plant variety data set with \( r \) replicates of \( m \) varieties in \( p \) environments (ordered accordingly).

We calculate the start value for \( \sigma^2 \) the same way we do for the variance components model, that is by using Henderson’s Method III.

\[
\hat{\sigma}^2 = \frac{SSE}{(n - r(W))}
\]

where \( SSE = y'y - y'W(W'W)^{-1}W'y \) and \( W = [X \ Z] \).

To obtain starting values for the elements of \( \gamma \), we need to formulate Henderson’s Method III for a multivariate model. Recall that the data \( y \) is ordered replicates within varieties within environments.

\[
y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_p \end{bmatrix}
\]

where \( y_i \) is the \( mr \times 1 \) vector of observations for the \( i \)th environment. The data can be presented as an \( mr \times p \) matrix

\[
Y = [y_1 \ y_2 \ \ldots \ y_p]
\]

such that \( y = \text{Vec}(Y) \). We can then write a multivariate linear mixed model

\[
Y = DT + BU + E
\]  \hspace{1cm} (9.12)

where \( D \) is an \( mr \times 1 \) design matrix, \( T = [\tau_1 \ \tau_2 \ \ldots \ \tau_p] \) is a \( 1 \times p \) matrix of environment means, \( B \) is an \( mr \times m \) design matrix, \( U = [u_1 \ u_2 \ \ldots \ u_p] \) is an \( m \times p \) matrix of \( m \) variety by \( p \) environment random effects, and \( E = [e_1 \ e_2 \ \ldots \ e_p] \) is an \( mr \times p \) matrix of residuals. By vectorising this multivariate linear mixed model, we can obtain the univariate form of the linear mixed model.

\[
\text{Vec}(Y) = \text{Vec}(DT) + \text{Vec}(BU) + \text{Vec}(E)
\]

\[
= (I_p \otimes D)\text{Vec}(T) + (I_p \otimes B)\text{Vec}(U) + \text{Vec}(E) \quad \text{(Using Result A.13)}
\]

\[
= (I_p \otimes D)\tau + (I_p \otimes B)u + e
\]

\[
= X\tau + Zu + e
\]

where \( X = (I_p \otimes D) \) and \( Z = (I_p \otimes B) \).
We now apply Henderson’s Method III to the multivariate linear mixed model by writing the model in a form that removes the distinction between fixed and random effects.

\[
Y = DT + BU + E = AS + E
\]

where \(A = \begin{bmatrix} D & B \end{bmatrix}\) and \(S = \begin{bmatrix} T \\ U \end{bmatrix}\).

In fitting a fixed effects model of the form \(Y = AS + E\), the estimate of the fixed effects is given by

\[
\hat{S} = (A' A)^{-1} A' Y
\]

as outlined in Mardia et al. (1979).

The residual sums of squares and products matrix after fitting \(Y = AS + E\) is

\[
SSE = \hat{E}' \hat{E} = (Y - A\hat{S})'(Y - A\hat{S}) = Y'Y - Y'A\hat{A} - \hat{A}'Y + \hat{A}'A\hat{A}
\]

\[
= Y'Y - Y'A(A' A)^{-1} A'Y - Y'A(A' A)^{-1} A' A(A' A)^{-1} A' Y
\]

\[
= Y'Y - Y'A(A' A)^{-1} A' Y
\]

Hence the reduction in the sums of squares and products matrix due to fitting \(Y = AS + E\) is

\[
R(S) = R(T, U) = Y'A(A' A)^{-1} A' Y
\]

When fitting the submodel \(Y = DT + E\), the reduction in the sums of squares and products matrix is

\[
R(T) = Y'D(D'D)^{-1} D'Y
\]

The reduction due to fitting \(U\) after accounting for \(T\) is

\[
R(U|T) = R(T, U) - R(T) = Y'A(A' A)^{-1} A'Y - Y'D(D'D)^{-1} D'Y
\]

To obtain estimates of the elements of \(\gamma\), we need to calculate

\[
E[R(U|T)] = E[Y'A(A' A)^{-1} A'Y] - E[Y'D(D'D)^{-1} D'Y]
\]
We need to find $E[Y'MY]$ where $M$ is any $mr \times mr$ matrix. First consider

$$Y'MY = \begin{bmatrix} y'_1 \\ y'_2 \\ \vdots \\ y'_p \end{bmatrix} M \begin{bmatrix} y_1 & y_2 & \cdots & y_p \end{bmatrix} = \begin{bmatrix} y'_1My_1 & y'_1My_2 & \cdots & y'_1My_p \\ y'_2My_1 & y'_2My_2 & \cdots & y'_2My_p \\ \vdots & \vdots & \ddots & \vdots \\ y'_pMy_1 & y'_pMy_2 & \cdots & y'_pMy_p \end{bmatrix}$$

Hence

$$Y'MY = \{y'_iM y_j\}$$

therefore

$$E[Y'MY] = \{E[y'_iM y_j]\}$$

where

$$E[y'_iM y_j] = E[\text{tr}(y'_iM y_j)] = E[\text{tr}(My_j y'_i)] = \text{tr}(ME[y_j y'_i])$$

Therefore we need to find $E[y_j y'_i]$.

Consider the multivariate mixed model $Y = DT + BU + E$ written as

$$\begin{bmatrix} y_1 & y_2 & \cdots & y_p \end{bmatrix} = \begin{bmatrix} D & \tau_1 & \tau_2 & \cdots & \tau_p \end{bmatrix} + \begin{bmatrix} B & u_1 & u_2 & \cdots & u_p \end{bmatrix} + \begin{bmatrix} e_1 & e_2 & \cdots & e_p \end{bmatrix}$$

Hence

$$y_i = D \tau_i + Bu_i + e_i = A \beta_i + e_i$$

where $A = [D \ B]$ and $\beta_i = \begin{bmatrix} \tau_i \\ u_i \end{bmatrix}$.

The data for environment $i$ has expectation

$$E[y_i] = AE[\beta_i]$$

and variance

$$\text{Var}[y_i] = A \text{Var}[\beta_i] A' + \text{Var}[e_i]$$

Consider

$$y_j y'_i = (A \beta_j + e_j)(A \beta_i + e_i)' = A \beta_j \beta_i' + A \beta_j e'_i + e_j \beta_i' A' + e_j e_i'$$
9.2. UNSTRUCTURED G MODEL AND RESULTS

Hence $y_jy_i'$ has expectation
\[
E[y_jy_i'] = AE[\beta_j\beta_i']A' + AE[\beta_j'e_i'] + E[e_je_i']
\]

Consider
\[
\beta_je_i' = \begin{bmatrix} \tau_j \\ u_j \end{bmatrix} e_i'
\]

hence the expectation of $\beta_je_i'$ is given by
\[
E[\beta_je_i'] = \begin{bmatrix} E[\tau_j e_i'] \\ E[u_j e_i'] \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}
\]

Therefore the expectation of $y_jy_i'$ is given by
\[
E[y_jy_i'] = AE[\beta_j\beta_i']A' + E[e_je_i']
\]

and hence
\[
E[Y'MY]_{ij} = E[y_jMy_i']
\]
\[
= \text{tr} \left( M( AE[\beta_j\beta_i']A' + E[e_je_i']) \right)
\]
\[
= \text{tr}(M\beta_j\beta_i'd) + \text{tr}(E[e_je_i'])
\]
\[
= \text{tr}(A'MAE[\beta_j\beta_i'] + \text{tr}(E[e_je_i'])
\]

Recall that to obtain estimates of the elements of $\gamma_i$ we need to calculate
\[
E[R(U|T)] = E[Y'A(A'A)^{-1}A'Y] - E[Y'D'(D'D)^{-1}D'Y]
\]

where
\[
E[Y'A(A'A)^{-1}A'Y]_{ij} = \text{tr}(A'A(A'A)^{-1}A'AE[\beta_j\beta_i']) + \text{tr}(A(A'A)^{-1}A'AE[e_je_i'])
\]
\[
= \text{tr}(A'A AE[\beta_j\beta_i']) + \text{tr}(A(A'A)^{-1}A'AE[e_je_i'])
\]

and
\[
E[Y'D'(D'D)^{-1}D'Y]_{ij} = \text{tr}(A'D'(D'D)^{-1}D'AE[\beta_j\beta_i']) + \text{tr}(D(D'D)^{-1}D'AE[e_je_i'])
\]

Therefore
\[
E[R(U|T)]_{ij} = E[Y'A(A'A)^{-1}A'Y] - E[Y'D'(D'D)^{-1}D'Y]
\]
\[
= \text{tr}(A'A AE[\beta_j\beta_i']) + \text{tr}(A(A'A)^{-1}A'AE[e_je_i'])
\]
\[
- \text{tr}(A'D'(D'D)^{-1}D'AE[\beta_j\beta_i']) - \text{tr}(D(D'D)^{-1}D'AE[e_je_i'])
\]
\[
= \text{tr}(A'A AE[\beta_j\beta_i']) - \text{tr}(A'D'(D'D)^{-1}D'AE[\beta_j\beta_i'])
\]
\[
+ \text{tr}(A(A'A)^{-1}A'AE[e_je_i']) - \text{tr}(D(D'D)^{-1}D'AE[e_je_i'])
\]
\[
= \text{tr}(A'M_D AE[\beta_j\beta_i'])
\]
\[
+ \text{tr}(A(A'A)^{-1}A'AE[e_je_i']) - \text{tr}(D(D'D)^{-1}D'AE[e_je_i'])
\]
\[
(9.14)
\]
where $M_D = I - D(D'D)^{-1}D'$.

To find the first trace term in (9.14), consider

$$AM_D A = \begin{bmatrix} D' \\ B' \end{bmatrix} M_D \begin{bmatrix} D & B \end{bmatrix}$$

$$= \begin{bmatrix} D'M_D D & D'M_D B \\ B'M_D D & B'M_D B \end{bmatrix}$$

where

$$D'M_D = D(I - D(D'D)^{-1}D')$$
$$= D' - D'D(D'D)^{-1}D'$$
$$= 0$$

hence

$$AM_D A = \begin{bmatrix} 0 & 0 \\ 0 & B'M_D B \end{bmatrix}$$

Consider

$$\beta_j \beta_i' = \begin{bmatrix} \tau_j \\ u_j \end{bmatrix} \begin{bmatrix} \tau_i \\ u_i' \end{bmatrix}$$

$$= \begin{bmatrix} \tau_j \tau_i & \tau_j u_i' \\ u_j \tau_i & u_j u_i' \end{bmatrix}$$

hence the expectation of $\beta_j \beta_i'$ is given by

$$E[\beta_j \beta_i'] = \begin{bmatrix} E[\tau_j \tau_i] & E[\tau_j u_i'] \\ E[u_j \tau_i] & E[u_j u_i'] \end{bmatrix}$$

$$= \begin{bmatrix} \tau_j \tau_i & 0 \\ 0 & \gamma_{ij} I_m \end{bmatrix}$$

Therefore

$$AM_D A E[\beta_j \beta_i'] = \begin{bmatrix} 0 & 0 \\ 0 & B'M_D B \end{bmatrix} \begin{bmatrix} \tau_j \tau_i & 0 \\ \gamma_{ij} B'M_D B \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 \\ \gamma_{ij} B'M_D B \end{bmatrix}$$

and hence

$$\text{tr}(A'M_D A E[\beta_j \beta_i']) = \gamma_{ij} \text{tr}(B'M_D B)$$

The expectation of $e_j e_i'$ is given by

$$E[e_j e_i'] = \begin{cases} \sigma^2 I_{mr}, & i = j \\ 0, & i \neq j \end{cases}$$
9.2. UNSTRUCTURED G MODEL AND RESULTS

hence, the second trace term in (9.14) is given by

\[
\tr(A(A'A)^{-1}A'[\epsilon_j \epsilon'_j]) = \begin{cases} 
\sigma^2 \tr(A(A'A)^{-1}A'), & i = j \\
0, & i \neq j 
\end{cases} 
\]

\[
= \begin{cases} 
\sigma^2 r(A), & i = j \\
0, & i \neq j 
\end{cases} 
\]

using Result A.15.

The third and final trace term in (9.14) is given by

\[
\tr(D(D'D)^{-1}D'[\epsilon_j \epsilon'_j]) = \begin{cases} 
\sigma^2 \tr(D'(D'D)^{-1}D'), & i = j \\
0, & i \neq j 
\end{cases} 
\]

\[
= \begin{cases} 
\sigma^2 r(D), & i = j \\
0, & i \neq j 
\end{cases} 
\]

using Result A.15.

Therefore

\[
E[R(U|T)]_{ij} = \begin{cases} 
\gamma_{ij}\tr(B'M_DB) + \sigma^2 (r(A) - r(D)), & i = j \\
\gamma_{ij}\tr(B'M_DB), & i \neq j 
\end{cases} 
\]

And hence

\[
E[R(U|T)] = \begin{bmatrix} 
\gamma_{11}\tr(B'M_DB) & \gamma_{12}\tr(B'M_DB) & \ldots & \gamma_{1p}\tr(B'M_DB) \\
\gamma_{21}\tr(B'M_DB) & \gamma_{22}\tr(B'M_DB) & \ldots & \gamma_{2p}\tr(B'M_DB) \\
\vdots & \vdots & \ddots & \vdots \\
\gamma_{p1}\tr(B'M_DB) & \gamma_{p2}\tr(B'M_DB) & \ldots & \gamma_{pp}\tr(B'M_DB) \\
\end{bmatrix} 
\]

\[
+ \begin{bmatrix} 
\sigma^2 (r(A) - r(D)) & 0 & \ldots & 0 \\
0 & \sigma^2 (r(A) - r(D)) & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \ldots & \sigma^2 (r(A) - r(D)) \\
\end{bmatrix} 
\]

\[
= \tr(B'M_DB) \begin{bmatrix} 
\gamma_{11} & \gamma_{12} & \ldots & \gamma_{1p} \\
\gamma_{21} & \gamma_{22} & \ldots & \gamma_{2p} \\
\vdots & \vdots & \ddots & \vdots \\
\gamma_{p1} & \gamma_{p2} & \ldots & \gamma_{pp} \\
\end{bmatrix} + \sigma^2 (r(A) - r(D)) \begin{bmatrix} 
1 & 0 & \ldots & 0 \\
0 & 1 & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \ldots & 1 \\
\end{bmatrix} 
\]

Thus the estimate of \( G_p \) is given by

\[
\hat{G}_p = \frac{R(U|T) - \hat{\sigma}^2 (r(A) - r(D)) I_p}{\tr(B'M_DB)} 
\]

where \( R(U|T) = Y'A(A'A)^{-1}A'Y - Y'D(D'D)^{-1}D'Y \).

This method can also be used to calculate starting values for unbalanced data if the data is expanded to a balanced data set and the missing values are imputed. There are numerous methods that can be used to impute the missing values. We compared replacing
the missing values with the overall mean, site mean and variety mean, as well as using the \texttt{norm} package in \textit{S-PLUS}. This software performs multiple imputation for multivariate continuous data under a normal model, and is described in Chapter 5 of Schafer (1997). We found that the starting values obtained using these various imputation methods were very similar and hence we have taken the simplest approach and have replaced the missing values with the overall mean of the response.

This method has been developed in the context of balanced multi-environment plant variety data. It can also be used for other types of multivariate data such as multi-trait animal breeding data. However, note that we have assumed that $R = \sigma^2 I_n$. As noted earlier, more complicated structures are often used for $R$ in the analysis of multi-environment plant variety trial and multi-trait animal breeding data. The extension of this method for more general $R$ structures is an area for possible future research.

\subsection*{9.2.7.2 Random Coefficient Data}

We calculate the start value for $\sigma^2$ the same way we do for the variance components model, that is by using Henderson’s Method III.

$$\hat{\sigma}^2 = \frac{SSE}{(n - r(W))}$$

where $SSE = y'y - y'W(W'W)^{-1}W'y$ and $W = [X \quad Z]$. However, we are not able to use Henderson’s Method III as we did above to obtain estimates of the random effects variance parameters.

Random coefficient regression models are used in situations where there are repeated measurements taken on $m$ subjects over $q$ times or experimental conditions. Polynomial response curves are fitted to the repeated measurements for each subject. The response curves vary randomly from subject to subject and are each defined by $p$ random regression coefficients. We assume that the data are ordered observations within subjects. The response variable $y$ can be presented as a $q \times m$ matrix

$$Y = [\begin{array}{c} y_1 \\ y_2 \\ \vdots \\ y_m \end{array}]$$

with a column for each subject, such that $y_i$ is the $q \times 1$ vector of repeated measurements over $q$ times for subject $i$. In this case, the multivariate linear mixed model that corresponds to the univariate linear mixed model $y = X\tau + Zu + e$ is of the form

$$Y = QT'D' + QU'B' + E$$

or

$$Y' = DTQ' + BUQ' + E' \tag{9.15}$$

where $D$ and $B$ are matrices consisting of zeros and ones, indicating the design across individuals, $Q$ is a regression matrix whose rows consist of powers of the $q$ time points. This model assumes that the number of coefficients is the same in the fixed and random part of the model but can be extended to a model in which fewer coefficients are fitted in
9.2. UNSTRUCTURED G MODEL AND RESULTS

The random part of the model than in the fixed part of the model. This type of model is commonly referred to as a growth curve model and was introduced by Potthoff and Roy (1964) and extended to include random effects as discussed in Lange and Laird (1989).

As we did for the multivariate data, we can write the model in a form that removes the distinction between fixed and random effects

\[
Y' = DTQ' + BUQ' + E' = ASQ' + E'
\]

where \( A = \begin{bmatrix} D & B \end{bmatrix} \) and \( S = \begin{bmatrix} T & U \end{bmatrix} \). In fitting a fixed effects model of the form \( Y' = ASQ + E' \), the estimate of the fixed effects is given by

\[
\hat{S} = (A'A)^{-1}A'YQ(Q'Q)^{-}Q'Y
\]

as outlined in Reinsel (1989). And hence the residual sums of squares and products matrix after fitting \( Y' = ASQ + E' \) is

\[
SSE = \hat{E}\hat{E}' = (Y' - A\hat{SQ}')(Y' - A\hat{SQ}')'
\]

\[
= Y'Y - Y'Q\hat{S}'A' - A\hat{SQ}'Y + A\hat{SQ}'Q\hat{S}'A'
\]

\[
= Y'Y - Y'Q(Q'Q)^{-}Q'YA(A'A)^{-}A' - A(A'A)^{-}A'Y'(Q'Q)^{-}Q'Y + A(A'A)^{-}A'Y'(Q'Q)^{-}Q'YA(A'A)^{-}A'
\]

This does not simplify easily like the residual sums of squares and products matrix in (9.13). Henderson’s Method III cannot be applied to this model as it was applied to the model in (9.12). The model in (9.15) has design matrices before and after the matrices of fixed and random effects. Hence we need to look at an alternative method for obtaining estimates of the elements of \( G_p \) (the random effects variance parameters) to use as start values for random coefficient models.

For the random coefficients model, the elements of \( G_p \) are the variances and covariances of the random deviations for each subject for each of the coefficients. If we fit the random effects as fixed effects, this will result in an estimate of each of the coefficients for each of the subjects. We can then obtain starting values for the random effect variance parameters by calculating the variance-covariance matrix for the estimated coefficients.
9.3 Random Coefficient Data Sets

9.3.1 Orthodontic Data

The first data set that we will look at is the Orthodontic data from Potthoff and Roy (1964). This data has been discussed by a number of authors including Hand and Crowder (1996), Verbeke and Molenberghs (1997), Pinheiro and Bates (2000) and Foulley and van Dyk (2000). Investigators at the University of North Carolina Dental School followed the growth of 27 children (16 males, 11 females) from age 8 until age 14. Every two years they measured the distance between the pituitary and the pterygomaxillary fissure, two points that are easily identified on x-ray exposures of the side of the head. The data are presented in Figure 9.1.

A possible model for the growth rate, allowing for gender differences, is a random coefficient model

\[ y_{ijk} = \alpha_i + \beta_i x_k + a_{ij} + b_{ij} x_k + e_{ijk} \]

where \( y_{ijk} \) is the distance from the centre of the pituitary to the pterygomaxillary fissure at age \( k \) (\( k = 8, 10, 12, 14 \)) for subject \( j \) (\( j = 1, \ldots, 27 \)) of gender \( i \) (\( i = 1, 2 \)). \( \alpha_i \) and \( \beta_i \) are the fixed intercept and slope for gender \( i \) and \( a_{ij} \) and \( b_{ij} \) are the random intercept and slope deviations for subject \( j \) who is of gender \( i \). This model allows the mean growth
pattern to be linear with age and dependent on the gender of the subject. Subjects have intercepts and slopes that vary about the gender-dependent averages.

In matrix notation we have

\[ y = X\tau + Zu + e \]

where \( y \) is the 108 \( \times \) 1 vector of facial growth measurements, ordered observations within subjects, \( \tau \) is the 4 \( \times \) 1 vector of fixed effects (male intercept and slope, female intercept and slope) and \( X \) is the 108 \( \times \) 4 associated design matrix, \( u \) is the 54 \( \times \) 1 vector of random effects (random intercept and slope deviations for all 27 subjects) and \( Z \) is the associated 108 \( \times \) 54 design matrix, and \( e \) is the 108 \( \times \) 1 vector of residuals. It is assumed that the joint distribution of \( u \) and \( e \) is given by

\[
\begin{bmatrix} u \\ e \end{bmatrix} \sim N \left( \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} G_2 \otimes I_{27} & 0 \\ 0 & \sigma^2 I_{108} \end{bmatrix} \right)
\]

where

\[
G_2 = \begin{bmatrix} \gamma_{11} & \gamma_{12} \\ \gamma_{21} & \gamma_{22} \end{bmatrix}
\]

such that \( \gamma_{11} \) is the variance of the intercept random deviations, \( \gamma_{22} \) is the variance of the slope random deviations, \( \gamma_{12} = \gamma_{21} \) is the covariance between the intercept and slope random deviations.

This model was fitted using the AI, EM and PXEM algorithms to estimate the variance parameters. The analyses were conducted using two different sets of starting values; uninformed starting values of \( G_p^{(0)} = I_2 \) and \( \sigma^2(0) = 1 \), and informed starting values calculated using the method outlined in Section 9.2.7.2. These values and the REML estimates are presented in Table 9.1.

Table 9.1: The uninformed and informed starting values and the REML estimates for the elements of \( G_p \) and \( \sigma^2 \), and the correlation \( \rho \) between the random intercept and slope for the Orthodontic data, rounded to 2 decimal places.

<table>
<thead>
<tr>
<th>Starting values</th>
<th>REML estimates</th>
<th>( \rho )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uninformed</td>
<td>Informed</td>
<td></td>
</tr>
<tr>
<td>( \gamma_{11} )</td>
<td>1.00</td>
<td>16.22</td>
</tr>
<tr>
<td>( \gamma_{12} )</td>
<td>0.00</td>
<td>-1.26</td>
</tr>
<tr>
<td>( \gamma_{22} )</td>
<td>1.00</td>
<td>0.14</td>
</tr>
<tr>
<td>( \sigma^2 )</td>
<td>1.00</td>
<td>1.72</td>
</tr>
</tbody>
</table>
The number of iterations to convergence for the AI, EM and PXEM algorithms and each set of starting values are given in Table 9.2. The results demonstrate the sensitivity of the AI algorithm to choice of starting values, with the AI algorithm failing when the uninformed starting values were used, yet converging in only 10 iterations when the informed starting values were used. In both cases the PXEM algorithm converged in far fewer iterations than the EM algorithm. While the PXEM algorithm outperformed the EM algorithm, it still fell short of the AI algorithm with informed starting values.

The AI algorithm, using the uninformed starting values, failed on the second iteration due to the AI variance parameter estimates being outside of the parameter space. Recall from Section 7.2 that we defined the parameter space as all values of the variance parameters such that the AI matrix is positive definite. Note that $G_p^{(1)}$ was not positive definite in the first iteration, resulting in $I_A(\kappa^{(2)})$ not being positive definite in the second iteration. Later in this chapter we will look at defining the parameter space as all values of the variance parameters such that $G_p$ is positive definite and $\sigma^2$ is positive.

Figure 9.2 displays the EM and PXEM iteration sequences for the elements of $G_p$ and $\sigma^2$ using the uninformed starting values. The PXEM estimates for $\gamma_{11}$ and $\gamma_{12}$ rapidly approached the REML solution within the first 5 iterations, whereas the EM estimates took over 100 iterations to get as close to the REML solution. However, the EM estimate for $\gamma_{22}$ approached the REML solution just as quickly as the PXEM estimates over the first 5 iterations. The PXEM estimate for $\sigma^2$ rapidly approached the REML solution within the first 5 iterations, whereas the EM estimate over shot the REML solution and then took over 100 iterations to return to the neighbourhood of the REML solution. Hence, overall the PXEM estimates rapidly approached the REML solutions within 5 iterations but then took another 45 iterations to converge. The EM estimates took over 100 iterations to get as close and then took over another 200 iterations to converge.

Table 9.2: The number of iterations to convergence for the AI, EM and PXEM algorithms for the Orthodontic data using the uninformed and informed starting values.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Starting values</th>
<th>Uninformed</th>
<th>Informed</th>
</tr>
</thead>
<tbody>
<tr>
<td>AI</td>
<td>fail</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>EM</td>
<td>339</td>
<td>271</td>
<td></td>
</tr>
<tr>
<td>PXEM</td>
<td>49</td>
<td>52</td>
<td></td>
</tr>
</tbody>
</table>

To be able to compare the performance of the AI algorithm to the EM and PXEM algorithms, we need to look at the results from the analyses using the informed starting values. The AI algorithm converged in 10 iterations using these starting values. It is interesting to note that the EM algorithm converged in fewer iterations using the informed starting values compared to the uninformed starting values, whereas the PXEM converged in a slightly greater number of iterations, indicating that our so called ”informed” starting values were not optimal.

Figure 9.3 displays the AI, EM and PXEM iteration sequences based on the informed
9.3. RANDOM COEFFICIENT DATA SETS

Figure 9.2: Iteration sequence for $\gamma_{11}$, $\gamma_{12}$, $\gamma_{22}$, and $\sigma^2$ for the EM and PXEM algorithms for the Orthodontic data using the uninformed starting values.

starting values for the elements of $G_P$ and $\sigma^2$. Similarly to Figure 9.2, the PXEM estimates for the elements of $G_P$ rapidly approached the REML solutions within 5 iterations but then took another 45 iterations to converge, and the EM estimates took over 100 iterations to get as close and then more than another 200 iterations to converge. The EM and PXEM estimates were closer to the REML solution than the AI estimates over the first few iterations, however once the AI estimates were close to the solution, the algorithm converged rapidly. The starting value obtained for $\sigma^2$ using Henderson’s Method III was equivalent to the REML estimate when rounded to 6 decimal places. The first PXEM iteration, and the first two EM iterations took $\sigma^2$ away from the REML estimate before heading back in the right direction, whereas the AI estimates of $\sigma^2$ remained in the neighbourhood of the solution.

The theoretical and empirical convergence rates for the AI, EM and PXEM algorithms, using both the uninformed and informed starting values, are presented in Table 9.3. The empirical rate of convergence is not presented for the AI algorithm with the uninformed starting values as the algorithm did not converge. The empirical convergence rates matched the theoretical convergence rates for all three algorithms, confirming that the algorithms did in fact converged. As expected, the convergence rate for the AI algorithm was approximately 0, indicating approximately quadratic convergence in the neighbourhood of the solution. The convergence rate for the EM algorithm was close to one indicating very slow convergence, and the convergence rate for the PXEM algorithm was lower than the convergence rate for the EM algorithm but still reasonably large indicating slow convergence.
CHAPTER 9. UNSTRUCTURED G MODEL

Figure 9.3: Iteration sequence for $\gamma_{11}$, $\gamma_{12}$, $\gamma_{22}$, and $\sigma^2$ for the AI, EM and PXEM algorithms for the Orthodontic data using the informed starting values.

Figure 9.4 shows the iteration sequence of $r_e^{(m)}$ obtained using the uninformed and informed starting values. The sequences of $r_e^{(m)}$ for the EM and PXEM algorithms using the informed starting values were monotone increasing, moving closer to the theoretical convergence rate with each iteration just as the estimates of the variance parameters in Figure 9.3 moved closer to the REML solution with each iteration. Whereas the sequences of $r_e^{(m)}$ for the EM and PXEM algorithms using the uninformed starting values started below the theoretical convergence rate and then jumped above it before slowly approaching it from above, just as the sequences for the estimates of the variance parameters in Figure 9.2 were not always monotonic.

The plot of the iteration sequences of $r_e^{(m)}$ obtained using the informed starting values, shows that the $r_e^{(m)}$ for the EM and PXEM algorithm was lower than that of the AI algorithm over the first few iterations. And as we saw in Figure 9.3, the AI estimates for the elements of $G_p$ were much further from the REML estimates than the EM and PXEM estimates over the early iterations. This suggests that the EM and PXEM algorithms were competitive in the early iterations and supports the use of a hybrid scheme that uses an EM type algorithm initially and then swaps to AI iterations once the variance parameter estimates are within the neighbourhood of the REML solution.

We will now look at using the various hybrid schemes to estimate the variance parameters. We need to select the cut-offs for $\Delta^{(m)}$ for the update criterion and for $p_{S^{(m)}}$ for the score criterion, and we need to decide how many internal local iterations to use for the local schemes. We will start by investigating the performance of the local hybrid schemes.
Table 9.3: The theoretical and empirical convergence rates for the AI, EM and PXEM algorithms for the Orthodontic data, rounded to 2 decimal places.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Theoretical</th>
<th>Empirical Uninformed</th>
<th>Empirical Informed</th>
</tr>
</thead>
<tbody>
<tr>
<td>AI</td>
<td>0.00</td>
<td>NA</td>
<td>0.00</td>
</tr>
<tr>
<td>EM</td>
<td>0.95</td>
<td>0.95</td>
<td>0.95</td>
</tr>
<tr>
<td>PXEM</td>
<td>0.73</td>
<td>0.73</td>
<td>0.73</td>
</tr>
</tbody>
</table>

Figure 9.4: The iteration sequence of $r_e^{(m)}$ for the EM and PXEM algorithms for the analysis of the Orthodontic data using the uninformed and informed starting values. For each algorithm, the dots represent $r_e^{(m)}$ at each iteration, and the horizontal line represents the theoretical convergence rate.

with varying numbers of internal local iterations using the AI criterion. Recall that the AI criterion uses a local iteration if an AI iteration produces estimates of the variance parameters that are outside the parameter space (i.e. result in a non-positive definite AI matrix). Using the uninformed starting values, the AI algorithm failed on the second iteration and so we well restart the iteration process with a local iteration.

Table 9.4 presents the number of iterations to convergence for the local EM/AI and local PXEM/AI schemes using from 1 to 10 internal local iterations. The local EM/AI scheme with 1-2 internal local iterations required two external local EM iterations before the variance parameter estimates were close enough to the REML solution to enable the AI algorithm to converge. Whereas when three or more internal local iterations were used, a single external local iteration was all that was needed to successfully switch to the AI algorithm. The local PXEM/AI scheme only required a single internal local iteration to obtain variance parameter estimates that were close enough the to REML solution to enable the AI algorithm to converge. We see that if too many internal local iterations were used for the local PXEM/AI scheme, it failed due to $A_v$ being singular and hence $\lambda^{(n+1)} = A_v^{(n)} b_v^{(n)}$ could not be calculated.

Figure 9.5 shows the iteration sequences for the elements of $G_p$ and $\sigma^2$ over the first 6
Table 9.4: The number of iterations to convergence for the analysis of the Orthodontic data using the local EM/AI and local PXEM/AI schemes with the AI criterion with from 1 to 10 internal local iterations, using the uninformed starting values.

<table>
<thead>
<tr>
<th>Number of internal local iterations</th>
<th>local EM/AI</th>
<th>local PXEM/AI</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9 (2/7)</td>
<td>8 (1/7)</td>
</tr>
<tr>
<td>2</td>
<td>8 (2/6)</td>
<td>9 (1/8)</td>
</tr>
<tr>
<td>3</td>
<td>7 (1/6)</td>
<td>10 (1/9)</td>
</tr>
<tr>
<td>4</td>
<td>7 (1/6)</td>
<td>10 (1/9)</td>
</tr>
<tr>
<td>5</td>
<td>7 (1/6)</td>
<td>10 (1/9)</td>
</tr>
<tr>
<td>6</td>
<td>7 (1/6)</td>
<td>10 (1/9)</td>
</tr>
<tr>
<td>7</td>
<td>7 (1/6)</td>
<td>fail</td>
</tr>
<tr>
<td>8</td>
<td>7 (1/6)</td>
<td>fail</td>
</tr>
<tr>
<td>9</td>
<td>7 (1/6)</td>
<td>fail</td>
</tr>
<tr>
<td>10</td>
<td>7 (1/6)</td>
<td>fail</td>
</tr>
</tbody>
</table>

iterations for the local EM/AI and local PXEM/AI schemes with varying numbers of internal local iterations. Note that the first two iterations were local iterations for the local EM/AI scheme with 1-2 internal local iterations, and the remaining iterations were AI iterations. Whereas only the first iteration was a local iteration for the local EM/AI schemes with more than 2 internal local iterations and for the local/PXEM schemes. This is reflected in the plots of the iteration sequences for $\sigma^2$, where the estimate of $\sigma^2$ was held at the starting value of 1 when a local iteration is done. The plots of the iteration sequences for $\gamma_{11}$ and $\gamma_{11}$ for the local EM/AI schemes indicate that 3 internal local iterations was the optimal number of internal local iterations for the local EM/AI scheme. The number of internal local iterations had very little effect on the iteration sequence for $\gamma_{12}$ for both the local EM/AI and local PXEM/AI schemes. We have chosen to use 3 internal local iterations for the local EM/AI scheme, and 1 internal local iteration for the local PXEM/AI scheme.
Figure 9.5: The iteration sequences for the elements of $G_p$ and $\sigma^2$ over the first 6 iterations using the local EM / AI and local PXEM / AI schemes with the AI criterion with different numbers of local iterations for the analysis of the Orthodontic data using the uninformed starting values. For each sequence, the points are labelled with the number corresponding to the number of internal local iterations used.

We will now select the cut-off for $\Delta^{(m)}$ for the update criterion for the hybrid schemes. Table 9.5 presents the results of the analyses of the Orthodontic data using the EM/AI scheme with the update criterion with the cut-off for $\Delta^{(m)}$ ranging from 0.5 to 1.5. When the cut-off was greater than or equal to 1.1, $\Delta^{(m)}$ was less than the cut-off at each iteration, however the AI estimates of the variance parameters were outside of the parameter space (i.e. the AI matrix was non-positive definite) on the 2nd, 3rd and 4th iterations and so EM iterations were used. Hence a cut-off of greater than or equal to 1.1 was not able to detect the convergence problems before they occurred. With a cut-off between 0.7 and 1.0, $\Delta^{(1)}$ was greater than the cut-off and the process was restarted with an EM iteration. The second iteration was then a successful AI iteration but the third AI iteration resulted in updates outside the parameter space, and so an EM iteration was done using the variance parameter estimates from the previous EM iteration. The remaining iterations were then successful AI iterations. Hence a cut-off for $\Delta^{(m)}$ between 0.7 and 1.0 was able to detect convergence problems before they occurred the first time but not the second time. When the cut-off was lowered to less than 0.7, the scheme was able to detect convergence problems before they occurred and invoked EM iterations from the beginning. However, with a cut-off less than 0.7, an unnecessarily large number of EM iterations were used before switching to AI iterations.

Recall that for the analysis of the incomplete block design data sets in Chapter 8, we used a cut-off of 1.1 for $\Delta^{(m)}$ as this value invoked EM type iterations only for the data sets were AI failed to converge, while detecting the convergence problems before they occurred. We will further investigate the selection of the cut-off for $\Delta^{(m)}$ for the update criterion.
for random coefficient models in Section 9.3.3 when we look at a range of simulated data sets. For the moment we will use a cut-off of 1.0 for $\Delta^{(m)}$ for the hybrid schemes with the update criterion for the analysis of the Orthodontic data.

Table 9.5: The sequence of iteration types for the Orthodontic data obtained using the EM/AI scheme with the update criterion with varying cut-offs for $\Delta^{(m)}$. "EM" denotes an EM iteration that was invoked due to $\Delta^{(m)} > \text{cut-off}$, and "EM*" denotes an EM iteration that was invoked due to the AI updates being outside the parameter space (i.e. the AI matrix was non-positive definite).

<table>
<thead>
<tr>
<th>Cut-off for $\Delta^{(m)}$</th>
<th>Iteration Sequence</th>
<th>Number of Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>19 EM, 5 AI</td>
<td>24</td>
</tr>
<tr>
<td>0.6</td>
<td>11 EM, 5 AI</td>
<td>16</td>
</tr>
<tr>
<td>0.7 - 1.0</td>
<td>EM, EM*, 6 AI</td>
<td>8</td>
</tr>
<tr>
<td>1.1 - 1.5</td>
<td>AI, 3 EM*, 6 AI</td>
<td>10</td>
</tr>
</tbody>
</table>

To use the hybrid schemes with the score criterion, we need to decide on the cut-off for $p_{S(m)}$. Table 9.6 presents the results of the analyses of the Orthodontic data using the EM/AI scheme with the score criterion using cut-offs for $p_{S(m)}$ ranging from 0.0001 to 0.5. When the cut-off was between 0.0001 and 0.1, $p_{S(m)}$ was less than the cut-off for the first 2 iterations, suggesting that the score vector was too far from 0 and hence the updates were too far from the REML solution, and so EM iterations were used. When the cut-off was increased to 0.5, EM iterations were used for the first three iterations due to $p_{S(m)}$ being less than the cut-off.

Table 9.6: The sequence of iterations to convergence for the score criterion EM/AI scheme with the cut-off for $p_{S(m)}$ ranging from 0.0001 to 0.5 for the Orthodontic data.

<table>
<thead>
<tr>
<th>Cut-off for $p_{S(m)}$</th>
<th>Iteration Sequence</th>
<th>Number of Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0001-0.1</td>
<td>2 EM, 6 AI</td>
<td>8</td>
</tr>
<tr>
<td>0.5</td>
<td>3 EM, 5 AI</td>
<td>8</td>
</tr>
</tbody>
</table>

Recall that for the incomplete block design data sets in Chapter 8, we chose a cut-off of 0.5 for $p_{S(m)}$. We will look at this further in Section 9.3.3 when we look at simulated data sets. For the moment, we will use a cut-off of 0.1 for $p_{S(m)}$ for the hybrid schemes with the score criterion for the analysis of the Orthodontic data.

Now that we have decided on the number of internal local iterations to use for the local schemes (3 for local EM/AI and 1 for local PXEM/AI) and the cut-offs for the update and score criteria for the hybrid schemes (use EM type iterations if $\Delta^{(m)} > 1.0$ for the update criterion and if $p_{S(m)} < 0.1$ for the score criterion), we will compare the results for the Orthodontic data for all of the hybrid schemes. The results are presented in Table 9.7.
Table 9.7: The number of iterations for the analysis of the Orthodontic data using the various hybrid schemes with the different criteria; 3 internal local iterations were used for the local EM/AI scheme, 1 internal local iteration was used for the local PXEM/AI scheme, a cut-off of 1.0 was used for $\Delta^{(m)}$ for the update criterion and a cut-off of 0.1 was used for $p_{S(m)}$ for the score criterion.

<table>
<thead>
<tr>
<th>Hybrid scheme</th>
<th>AI criterion</th>
<th>Update criterion</th>
<th>Score criterion</th>
</tr>
</thead>
<tbody>
<tr>
<td>EM/AI</td>
<td>8 (2/6)</td>
<td>8 (2/6)</td>
<td>8 (2/6)</td>
</tr>
<tr>
<td>PXEM/AI</td>
<td>7 (2/5)</td>
<td>7 (2/5)</td>
<td>7 (2/5)</td>
</tr>
<tr>
<td>local EM/AI</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
<td>fail</td>
</tr>
<tr>
<td>local PXEM/AI</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td>fail</td>
</tr>
</tbody>
</table>

The results for the hybrid schemes with the update and score criteria were equivalent to the results obtained using the AI criterion for the EM/AI and PXEM/AI schemes, indicating that both the criterion successfully detected convergence problems before they occurred. Note that the AI algorithm failed on the second iteration due to the AI updates being outside the parameter space (i.e. the variance parameter estimates resulted in a non-positive definite AI matrix). The EM/AI and PXEM/AI schemes with the update and score criteria were able to detect this on the first iteration and invoked an EM type iteration. However note that the second EM and PXEM iterations used in the EM/AI and PXEM/AI schemes with the update criterion were invoked due to the AI iteration producing variance parameter estimates outside the parameter space rather than due to $p_{S(m)} < 0.1$. Whereas both of the EM and PXEM iterations used in the EM/AI and PXEM/AI schemes with the score criterion were invoked due to $p_{S(m)} < 0.1$. Hence the update criterion was better at detecting convergence problems before they occurred.

The results for the local EM/AI and local PXEM/AI schemes for the update criterion matched the results for the AI criterion. However the local schemes using the score criterion failed as the iteration sequences consisted of entirely local iterations. The local iterations only updated the elements of $G_p$ and held $\sigma^2 = 1$. Holding $\sigma^2 = 1$ resulted in $p_{S(m)}$ being less than 0.1 for all iterations and hence the algorithm was unable to switch to AI iterations. Hence the update criterion needs to be used for the local hybrid schemes for the analysis of the Orthodontic data.

Finally, Table 9.8 provides a summary of the number of iterations to convergence for the Orthodontic data for the AI, EM and PXEM algorithms, and the hybrid schemes with the update criterion with a cut-off of 1.0 for $\Delta^{(m)}$. The hybrid schemes performed well, all converging in 7 or 8 iterations, far fewer than the EM and PXEM algorithms. The hybrid schemes also outperformed the AI algorithm with informed starting values, however as mentioned previously, the informed starting values used here were not optimal due to the difficulty in obtaining good starting values for random coefficient models. These results indicate that the hybrid schemes were robust to choice of starting values yet still were able to converge quickly.
### Table 9.8: A summary of the number of iterations to convergence for the Orthodontic data using the uninformed starting values for the AI, EM, PXEM algorithms as well as the EM/AI, PXEM/AI, local EM/AI and local PXEM/AI schemes with the update criterion with a cut-off of 1.0 for $\Delta^{(m)}$ using the uninformed starting values, and for the AI algorithm using the informed starting values.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Number of Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>AI</td>
<td>fail</td>
</tr>
<tr>
<td>EM</td>
<td>339</td>
</tr>
<tr>
<td>PXEM</td>
<td>49</td>
</tr>
<tr>
<td>EM/AI</td>
<td>8 (2/6)</td>
</tr>
<tr>
<td>PXEM/AI</td>
<td>7 (2/5)</td>
</tr>
<tr>
<td>local EM/AI</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td>local PXEM/AI</td>
<td>8 (1/8)</td>
</tr>
<tr>
<td>AI (informed)</td>
<td>10</td>
</tr>
</tbody>
</table>

### 9.3.2 Ultrafiltration Data

The next data set that we will look at is the Ultrafiltration data from Vonesh and Carter (1992). This data has been discussed by a number of authors including Vonesh and Chinchilli (1997), Pinheiro and Bates (2000) and Foulley and van Dyk (2000).

Standard low flux membrane dialyzers, which are used in hemodialysis to treat patients with end-stage renal disease, have their water transport kinetics characterized by a linear relationship between the ultrafiltration rate at which water is removed and the transmembrane pressure that is exerted on the dialyzer membrane (Vonesh and Carter, 1987). More recently, high flux membrane dialyzers have been introduced for the purpose of reducing the time spent by patients on hemodialysis. Unlike their low flux counterparts, high flux dialyzers have their water transport kinetics characterized by a nonlinear relationship between ultrafiltration rate and transmembrane pressure.

Vonesh and Carter (1992) analysed data measured on high-flux hemodialysers to assess their in vivo ultrafiltration characteristics. The ultrafiltration rates (in ml/hr) of 20 high flux dialyzers were measured at seven ascending transmembrane pressures (in mmHg). The dialyzers were evaluated in vitro using bovine blood at flow rates of either 200 ml/min or 300 ml/min. The data are presented in Figure 9.6.

We consider the random coefficient model for ultrafiltration rate as presented in Foulley and van Dyk (2000),

$$ y_{ijk} = \alpha_i + \beta_{1,i} x_{ijk} + \beta_{2,i} x_{ijk}^2 + \beta_{3,i} x_{ijk}^3 + \beta_{4,i} x_{ijk}^4 + a_{ij} + b_{1,ij} x_{ijk} + b_{2,ij} x_{ijk}^2 + e_{ijk} $$

where $y_{ijk}$ is the ultrafiltration rate at the $k$th transmembrane pressure for the $j$th dialyzer ($j = 1, \ldots, 20$) at blood flow rate $i$ ($i = 1, 2$). $\alpha_i$, $\beta_{1,i}$, $\beta_{2,i}$, $\beta_{3,i}$, $\beta_{4,i}$ are the fixed coefficients for blood flow rate $i$, $a_{ij}$, $b_{1,ij}$ and $b_{2,ij}$ are the random deviations for the first
three coefficients for dialyzer $j$ at blood flow rate $i$, and $x_{ijk}$ is the $k$th transmembrane pressure for the $j$th dialyzer at blood flow rate $i$.

In matrix form we have

$$ y = X\tau + Zu + e $$

where $y$ is the $140 \times 1$ vector of ultrafiltration rates, $\tau$ is the $10 \times 1$ vector of fixed effects and $X$ is the associated $140 \times 10$ design matrix of full column rank, $u = (u_0', u_1', u_2')$ is the $60 \times 1$ vector of random effects partitioned into the dialyzer intercept, linear and quadratic coefficients and $Z$ is the associated $140 \times 60$ design matrix, and $e$ is the $140 \times 1$ vector of residuals. It is assumed that the joint distribution of $u$ and $e$ is given by

$$ \begin{bmatrix} u \\ e \end{bmatrix} \sim N \left( \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} G \otimes I_{20} & 0 \\ 0 & \sigma^2 I_{140} \end{bmatrix} \right) $$

where

$$ G_p = \begin{bmatrix} \gamma_{11} & \gamma_{12} & \gamma_{13} \\ \gamma_{21} & \gamma_{22} & \gamma_{23} \\ \gamma_{31} & \gamma_{32} & \gamma_{33} \end{bmatrix} $$

This model was fitted using the AI, EM and PXEM algorithms to estimate the variance parameters. The analyses were conducted using two different sets of starting values;
uninformed starting values of $G_p^{(0)} = I_2$ and $\sigma^2(0) = 1$, and informed starting values calculated using the method described in Section 9.2.7.2 These values and the REML estimates are presented in Table 9.9.

Table 9.9: The uninformed and informed starting values and the REML estimates for the elements of $G$ and $\sigma^2$, and the correlations $\rho$ between the random coefficients for the Ultrafiltration data analysis, rounded to 2 decimal places.

<table>
<thead>
<tr>
<th></th>
<th>Starting values</th>
<th>REML estimates</th>
<th>$\rho$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Uninformed</td>
<td>Informed</td>
<td>2.25</td>
</tr>
<tr>
<td>$\gamma_{11}$</td>
<td>1.00</td>
<td>9.30</td>
<td>-0.51</td>
</tr>
<tr>
<td>$\gamma_{12}$</td>
<td>0.00</td>
<td>-13.73</td>
<td>-3.73</td>
</tr>
<tr>
<td>$\gamma_{13}$</td>
<td>0.00</td>
<td>2.00</td>
<td>0.69</td>
</tr>
<tr>
<td>$\gamma_{22}$</td>
<td>1.00</td>
<td>39.82</td>
<td>24.08</td>
</tr>
<tr>
<td>$\gamma_{23}$</td>
<td>0.00</td>
<td>-8.57</td>
<td>-6.83</td>
</tr>
<tr>
<td>$\gamma_{33}$</td>
<td>1.00</td>
<td>3.07</td>
<td>2.17</td>
</tr>
<tr>
<td>$\sigma^2$</td>
<td>1.00</td>
<td>3.33</td>
<td>3.32</td>
</tr>
</tbody>
</table>

The number of iterations to convergence for the AI, EM and PXEM algorithms and each set of starting values are given in Table 9.10. The AI algorithm failed for both sets of starting values. Note that the AI algorithm also failed when started from the REML estimates of the variance parameters rounded to integers indicating that the AI algorithm only performs well in a very small neighbourhood of the REML solution. Once again we see that the PXEM algorithm converged in far fewer iterations than the EM algorithm. Figures 9.7 and 9.8 display the iteration sequences for the elements of $G_p$ and $\sigma^2$ for the EM and PXEM algorithms based on the uninformed and informed starting values, respectively. The PXEM estimates of the variance parameters approached the REML solutions more rapidly than the EM estimates, particularly for the uninformed starting values.

Table 9.10: The number of iterations to convergence for the AI, EM and PXEM algorithms for the Ultrafiltration data using the uninformed and informed starting values.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Starting values</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Uninformed</td>
</tr>
<tr>
<td>AI</td>
<td>fail</td>
</tr>
<tr>
<td>EM</td>
<td>263</td>
</tr>
<tr>
<td>PXEM</td>
<td>40</td>
</tr>
</tbody>
</table>

The theoretical and empirical convergence rates for the AI, EM and PXEM algorithms are presented in Table 9.11. The empirical convergence rates were calculated from the iteration sequence obtained using the uninformed starting values. The empirical rate of convergence was not available for the AI algorithm as it did not converge. The empirical convergence rates equaled the theoretical convergence rates for the EM and PXEM algorithms, confirming that the algorithms did converged. As expected, the convergence rate
Figure 9.7: Iteration sequence for the elements of $G_p$ and $\sigma^2$ for the EM and PXEM algorithms for the Ultrafiltration data using the uninformed starting values.
Figure 9.8: Iteration sequence for the elements of $G_p$ and $\sigma^2$ for the EM and PXEM algorithms for the Ultrafiltration data using the informed starting values.
Table 9.11: The theoretical and empirical convergence rates for the AI, EM and PXEM algorithms for the Ultrafiltration data, rounded to 2 decimal places.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Theoretical</th>
<th>Empirical</th>
</tr>
</thead>
<tbody>
<tr>
<td>AI</td>
<td>0.01</td>
<td>NA</td>
</tr>
<tr>
<td>EM</td>
<td>0.94</td>
<td>0.94</td>
</tr>
<tr>
<td>PXEM</td>
<td>0.67</td>
<td>0.67</td>
</tr>
</tbody>
</table>

for the AI algorithm was approximately 0, indicating approximately quadratic convergence in the neighbourhood of the solution. The convergence rate for the EM algorithm was close to 1 indicating very slow convergence, and the convergence rate for the PXEM algorithm was lower than the convergence rate for the EM algorithm but still reasonable large indicating slow convergence.

Figure 9.9 shows the iteration sequence of $r_e^{(m)}$ for the EM and PXEM algorithms obtained using both the uninformed and informed starting values. As the iteration number increased, $r_e^{(m)}$ stabilised at the theoretical convergence rate in each of the plots, indicating that the algorithms did converge. Note that $r_e^{(m)}$ was low for both the EM and PXEM algorithms in the early iterations due to the EM and PXEM estimates of the variance parameter estimates rapidly approaching the REML solution in the early iterations as seen in Figures 9.7 and 9.8. This indicates that the EM and PXEM algorithms were competitive in the early iterations and thus supports the use of a hybrid scheme that uses EM or PXEM iterations initially and then switches to AI iterations.

Figure 9.9: The iteration sequence of the empirical convergence rate for the EM and PXEM algorithms with uninformed and informed starting values for the Ultrafiltration data.

We will now look at using the various hybrid schemes to analyse this data set. We need to select the cut-offs for $\Delta_e^{(m)}$ for the update criterion and for $p_{S^{(m)}}$ for the score criterion, and we need to decide how many internal local iterations to use for the local schemes. We will start by investigating the performance of the local hybrid schemes with varying numbers of internal local iterations using the AI criterion. Using the uninformed starting values, the AI algorithm failed on the second iteration due to the AI updates being outside the
parameter space (i.e. the variance parameter estimates resulted in a non-positive definite AI matrix), and so the local hybrid schemes will restart the iteration process with a local iteration.

Table 9.12 presents the number of iterations to convergence for the local EM/AI and local PXEM/AI schemes using from 1 to 10 internal local iterations. All of the schemes only required a single local iteration before successfully switching to AI iterations and they all converged in the same number of iterations.

Table 9.12: The number of iterations to convergence for the analysis of the Ultrafiltration data using the local EM/AI and local PXEM/AI schemes with the AI criterion with from 1 to 10 internal local iterations.

<table>
<thead>
<tr>
<th>Number of internal local iterations</th>
<th>local EM/AI</th>
<th>local PXEM/AI</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10 (1/9)</td>
<td>10 (1/9)</td>
</tr>
<tr>
<td>2</td>
<td>10 (1/9)</td>
<td>10 (1/9)</td>
</tr>
<tr>
<td>3</td>
<td>10 (1/9)</td>
<td>10 (1/9)</td>
</tr>
<tr>
<td>4</td>
<td>10 (1/9)</td>
<td>10 (1/9)</td>
</tr>
<tr>
<td>5</td>
<td>10 (1/9)</td>
<td>10 (1/9)</td>
</tr>
<tr>
<td>6</td>
<td>10 (1/9)</td>
<td>10 (1/9)</td>
</tr>
<tr>
<td>7</td>
<td>10 (1/9)</td>
<td>10 (1/9)</td>
</tr>
<tr>
<td>8</td>
<td>10 (1/9)</td>
<td>10 (1/9)</td>
</tr>
<tr>
<td>9</td>
<td>10 (1/9)</td>
<td>10 (1/9)</td>
</tr>
<tr>
<td>10</td>
<td>10 (1/9)</td>
<td>10 (1/9)</td>
</tr>
</tbody>
</table>

To get a closer look at these schemes, the estimates of the elements of $G_p$ and $\sigma^2$ over the first 6 iterations are plotted in Figure 9.10. The plots suggest that the local EM/AI schemes that used at least four internal local EM iterations produced variance parameter estimates in the first iteration that were closer to the REML solution than the estimates from the schemes that used less than four internal local iterations. The plots indicate that the local PXEM/AI scheme that used a single internal local EM iterations produced variance parameter estimates in the first iteration that were closer to the REML solution than the estimates from the local PXEM/AI schemes with more than one internal local iterations.

Although the plots in Figure 9.10 indicate that 4 internal local iterations took the variance parameter estimates closer to the REML solution, the results in Table 9.12 indicate that the local EM scheme with a single internal local iteration converged in the same number of iterations as the scheme with 4 internal local iterations. There is no need to do the extra 3 internal local iterations and so for the analysis of the Ultrafiltration data we will use a single internal local iteration for the local EM/AI scheme. For the local PXEM/AI scheme, we will also use a single internal local iteration.

We now select the cut-off for $\Delta^{(m)}$ for the update criterion for the hybrid schemes. Table 9.13 presents the results of the analyses of the Ultrafiltration data using the EM/AI scheme with the update criterion with the cut-off for $\Delta^{(m)}$ ranging from 0.5 to 1.5. The results suggest that the optimal cut-off lies between 0.7 and 0.9, and so for the analysis of
9.3. RANDOM COEFFICIENT DATA SETS

Figure 9.10: The iteration sequences for the elements of $G_p$ and $\sigma^2$ over the first 6 iterations using the local EM/AI and local PXEM/AI schemes with the AI criterion with different numbers of internal local iterations for the Ultrafiltration data using the uninformed starting values. For each sequence, the points are labelled with the number corresponding to the number of internal local iterations used.
the Ultrafiltration data using the hybrid schemes with the update criterion, we will use a cut-off of 0.9 for $\Delta^{(m)}$.

Table 9.13: The sequence of iterations to convergence for the Ultrafiltration data obtained using the EM/AI scheme with the update criterion with a cut-off for $\Delta^{(m)}$ ranging from 0.5 to 1.5. "EM" denotes an EM iteration that was invoked due to $\Delta^{(m)} > \text{cut-off}$, and "EM*" denotes an EM iteration that was invoked due to the AI updates being outside of the parameter space (i.e. the AI matrix was non-positive definite).

<table>
<thead>
<tr>
<th>Cut-off for $\Delta^{(m)}$</th>
<th>Iteration Sequence</th>
<th>Number of Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>8 EM, 7 AI</td>
<td>15</td>
</tr>
<tr>
<td>0.6</td>
<td>5 EM, 8 AI</td>
<td>13</td>
</tr>
<tr>
<td>0.7-0.9</td>
<td>EM, 10 AI</td>
<td>11</td>
</tr>
<tr>
<td>1.0-1.5</td>
<td>AI, 3 EM*, 9 AI</td>
<td>13</td>
</tr>
</tbody>
</table>

Table 9.14 presents the results of the analyses of the Ultrafiltration data using the EM/AI scheme with the score criterion using cut-offs ranging from 0.0001 to 0.5. The results indicate that a cut-off less than or equal to 0.1 was optimal for the score criterion. However, although the final iteration sequences were the same for all of the schemes with a cut-off less than or equal to 0.1, the actual sequences were not identical. When the cut-off was between 0.05 and 0.1, EM iterations were invoked for the first two iterations due to the attempted AI iterations resulting in $p_{S(m)}$ less than the cut-off. When the cut-off was less than or equal to 0.01, an EM iteration was invoked for the first iteration due $p_{S(m)}$ less than the cut-off. But then two AI iterations were done before $p_{S(m)}$ was less than the cut-off again and the iteration process was resumed with an EM iteration using the variance parameter estimates from the previous EM iteration. And so it is quicker to use a cut-off between 0.05 and 0.1. For the analysis of the Ultrafiltration data we will use a cut-off of 0.1.

Table 9.14: The sequence of iterations to convergence for the score criterion EM/AI scheme with the cut-off for $p_{S(m)}$ ranging from 0.5 to 1.5 for the Ultrafiltration data.

<table>
<thead>
<tr>
<th>Cut-off for $p_{S(m)}$</th>
<th>Iteration Sequence</th>
<th>Number of Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0001 - 0.1</td>
<td>2 EM, 9 AI</td>
<td>11</td>
</tr>
<tr>
<td>0.5</td>
<td>5 EM, 8 AI</td>
<td>13</td>
</tr>
</tbody>
</table>

Now that we have decided on the number of internal local iterations to use for the local schemes (1 for local EM/AI and 1 for local PXEM/AI) and the cut-offs for the update and score criteria for the hybrid schemes (use EM type iterations if $\Delta^{(m)} > 0.9$ for the update criterion and if $p_{S(m)} < 0.1$ for the score criterion), we will compare the results for the Ultrafiltration data for all of the hybrid scheme. The results are presented in Table 9.15.
Table 9.15: The number of iterations for the analysis of the Ultrafiltration data using the various hybrid schemes with the different criteria; 2 internal local iterations were used for the local EM/AI scheme, 1 internal local iteration was used for the local PXEM/AI scheme, a cut-off of 0.9 was used for $\Delta^{(m)}$ for the update criterion and a cut-off of 0.1 was used for $p_{S^{(m)}}$ for the score criterion.

<table>
<thead>
<tr>
<th>Hybrid scheme</th>
<th>AI criterion</th>
<th>Update criterion</th>
<th>Score criterion</th>
</tr>
</thead>
<tbody>
<tr>
<td>EM/AI</td>
<td>11 (1/10)</td>
<td>11 (1/10)</td>
<td>11 (2/9)</td>
</tr>
<tr>
<td>PXEM/AI</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
</tr>
<tr>
<td>local EM/AI</td>
<td>10 (1/9)</td>
<td>10 (1/9)</td>
<td>fail</td>
</tr>
<tr>
<td>local PXEM/AI</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td>fail</td>
</tr>
</tbody>
</table>

The results for the hybrid schemes with the update criterion were the same as the results for the AI criterion hybrid schemes, indicating that the update criterion successfully identified convergence problems before they occurred. The hybrid schemes with the AI criterion restarted with an EM type iteration after the AI algorithm failed on the second iteration due to the AI updates being outside the parameter space (i.e. the AI matrix was non-positive definite). The hybrid schemes with the update criterion were able to detect this on the first iteration and invoked an EM iteration before the convergence problems occurred. The hybrid schemes with the score criterion also detected this on the first iteration. However the score criterion invoked two EM iterations for the EM/AI scheme, compared to only one with the update criterion. Note that the total number of iterations to convergence was the same for both criteria. The results of the score criterion PXEM/AI scheme were equivalent to the update criterion PXEM/AI scheme.

The local schemes with the score criterion failed. The iteration sequences consisted entirely of local iterations, only updating the elements of $G_p$ and holding $\sigma^2 = 1$ and hence did not converge to the REML estimates. Holding $\sigma^2 = 1$ resulted in the score function for $\sigma^2$ being large, resulting in a large value for $S^{(m)}$ and hence $p_{S^{(m)}}$ was less than 0.1 for all iterations and the algorithm was unable to switch to AI iterations. If the cut-off for the score criterion was lowered to less than $1^{-10}$ then the algorithm was able to switch to AI iterations. However, such a low cut-off for $p_{S^{(m)}}$ failed to invoke EM type iterations before an AI iteration fails for other data sets, as we will see later. Alternatively, if the starting value for $\sigma^2$ was closer to the REML estimate, the local schemes were able to switch back to AI iterations after a single local iteration. We need a hybrid scheme that is robust to poor starting values, hence the update criterion needs to be used for the hybrid schemes for the analysis of the Ultrafiltration data.

Finally, Table 9.16 provides a summary of the number of iterations to convergence for the Ultrafiltration data for the AI, EM and PXEM algorithms, and the hybrid schemes with the update criterion with a cut-off of 0.9 for $\Delta^{(m)}$ using the uninformed starting values, and for the AI algorithm using the informed starting values. The AI algorithm converged using both the informed and uninformed starting values. However, as previously mentioned, the informed starting values used here were not optimal due to the difficulty in obtaining good starting values for random coefficient models. The hybrid schemes converged in far fewer iterations than the EM and PXEM algorithms, with the PXEM/AI and local PXEM/AI schemes outperforming the EM/AI and local EM/AI schemes, but only by 2
iterations. These results highlight the sensitivity of the AI algorithm to choice of starting values and support the use of a hybrid scheme.

Table 9.16: The number of iterations to convergence for the Ultrafiltration data using the uninformed starting values for the AI, EM and PXEM algorithms and the EM/AI, PXEM/AI, local EM/AI and local PXEM/AI schemes with the update criterion with a cut-off of 0.9 for \( \Delta^{(m)} \), and for the AI algorithm using the informed starting values.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Number of Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>AI</td>
<td>fail</td>
</tr>
<tr>
<td>EM</td>
<td>263</td>
</tr>
<tr>
<td>PXEM</td>
<td>40</td>
</tr>
<tr>
<td>EM/AI</td>
<td>11 (1/10)</td>
</tr>
<tr>
<td>PXEM/AI</td>
<td>8 (1/7)</td>
</tr>
<tr>
<td>local EM/AI</td>
<td>10 (1/9)</td>
</tr>
<tr>
<td>local PXEM/AI</td>
<td>8 (1/7)</td>
</tr>
<tr>
<td>AI (informed)</td>
<td>fail</td>
</tr>
</tbody>
</table>

9.3.3 Simulated Random Coefficient Data

We will now investigate the performance of the various schemes for some simulated random coefficient data sets. Data were simulated based on a scenario where repeated measurements were taken on 20 subjects over 5 times. Each subject was assigned to one of 2 treatments with 10 subjects in each treatment group. The data were simulated as though the response curve for each subject followed a linear trend with an intercept and slope that randomly deviated from the group intercepts and slopes. If \( y \) is the 100 \( \times \) 1 vector of observations, ordered observations within subjects, then the model is given by

\[
y = X\tau + Zu + e
\]

where \( \tau \) is the 4\( \times \)1 vector of fixed effects consisting of the intercept and slope for treatment groups 1 and 2, and \( X \) is the 100 \( \times \) 4 associated design matrix. \( u \) is the 40 \( \times \) 1 vector of random effects consisting of the random intercept and slope deviations for each of the 20 subjects, and \( Z \) is the 100 \( \times \) 40 associated design matrix. \( e \) is the 100 \( \times \) 1 vector of residuals. We assume that the joint distribution of \( u \) and \( e \) is given by

\[
\begin{bmatrix}
  u \\
  e
\end{bmatrix}
\sim N
\left(
\begin{bmatrix}
  0 \\
  0
\end{bmatrix},
\begin{bmatrix}
  G_2 \otimes I_{20} & 0 \\
  0 & \sigma^2 I_{100}
\end{bmatrix}
\right)
\]

where

\[
G_2 = \begin{bmatrix}
  \gamma_{11} & \gamma_{12} \\
  \gamma_{21} & \gamma_{22}
\end{bmatrix}
\]

The first 10 data sets were simulated using values of \( \gamma = (1,0,1)' \) and \( \sigma^2 = 1 \), the second 10 data sets were simulated using values of \( \gamma = (1,0.5,1)' \) and \( \sigma^2 = 1 \), and the last 10
data sets were simulated using values of $\gamma = (1, 0.8, 1)'$ and $\sigma^2 = 1$. The number of iterations to convergence for the AI, EM and PXEM algorithms, using starting values of $\gamma^{(0)} = (1, 0, 1)'$ and $\sigma^{2(0)} = 1$, are given in Table 9.17. The REML estimates are also presented. Note that the iteration sequence was terminated if the algorithm did not converge within 500 iterations. This is represented by " > 500" in Table 9.17.

The AI algorithm converged for 9 of the 10 data sets in setting 1, 8 of the 10 data sets simulated in setting 2, and 5 of the 10 data sets in setting 3. Hence the stronger the correlation between the random intercepts and slopes, the poorer the performance of the AI algorithm. However, for the data sets where the AI algorithm converged, it clearly outperformed the EM and PXEM algorithms. The AI algorithm converged in less than half the number of iterations of the PXEM algorithm and at least one fifth of the number of iterations of the EM algorithm.

Rather than looking at the iteration sequences for each of the variance parameters for each of the algorithms for each of the data sets, we will just look at the iteration sequences of $r_e^{(m)}$ for each of the algorithms. Figures 9.11, 9.12 and 9.13 display the iteration sequence of $r_e^{(m)}$ and the theoretical convergence rate for the AI, EM and PXEM algorithms for the random coefficient data sets simulated using values of $\gamma = (1, 0, 1)'$, $\gamma = (1, 0.5, 1)'$ and $\gamma = (1, 0.8, 1)'$ respectively. We expect $r_e^{(m)}$ to approach the theoretical convergence rate as $m$ increases.

For the data sets where the AI algorithm converged, we see that $r_e^{(m)}$ approached the theoretical convergence rate over the first 5 or so iterations and then rapidly stabilised within a couple of iterations. For the PXEM algorithm, $r_e^{(m)}$ also approached the theoretical convergence rate over the first 5 or so iterations for most data sets, but then took at least 20 iterations to stabilise to the theoretical convergence rate. For the EM algorithm, $r_e^{(m)}$ took at least 20 iterations to approach the theoretical convergence rate for most of the data sets, and then at least 40 iterations to converge. Note that for the data sets for which the EM algorithm does not converge within 500 iterations, $r_e^{(m)}$ was close to but not yet equal to the theoretical rate at the 500th iteration. For most of the data sets we see that $r_e^{(m)}$ for the EM and PXEM algorithms was lower than that of the AI algorithm over the first few iterations, indicating that the EM and PXEM algorithms were competitive in the early iterations. However, once the variance parameter estimates were in the neighbourhood of the REML solution, the AI algorithm converged rapidly. The EM and PXEM updates rapidly approached the REML solution in the early iterations, but then took very small steps once in the neighbourhood of the solution and converged much more slowly than the AI algorithm.

Table 9.18 contains the theoretical and empirical convergence rates for the AI, EM and PXEM algorithms. The empirical convergence rate for the cases where the AI algorithm did not converge are presented as NAs. For the data sets where the EM algorithm did not converge within 500 iterations, $r_e^{(500)}$ is presented in place of the empirical convergence rate. Where available, the empirical convergence rates were equal to the theoretical convergence rate for all algorithms for all data sets. The theoretical convergence rate for the AI algorithm was approximately zero for all data sets, including those for which the AI algorithm did not converge. This indicates that the AI algorithm has approximate quadratic convergence in the neighbourhood of the REML solution. The convergence rate
Table 9.17: The number of iterations to convergence for the AI, EM and PXEM algorithms for the simulated random coefficient data sets using the uninformed starting values; $G_p^{(0)} = I_2$ and $\sigma^2(0) = 1$. The REML estimates for the elements of $G_p$ and $\sigma^2$, and the correlation $\rho$ between the random intercept and slope are also presented, rounded to 2 decimal places. The data sets in setting 1 were simulated using $\gamma = (1, 0, 1)'$ and $\sigma^2 = 1$, the data sets in setting 2 were simulated using values of $\gamma = (1, 0.5, 1)'$ and $\sigma^2 = 1$, and the data sets in setting 3 were simulated using values of $\gamma = (1, 0.8, 1)'$ and $\sigma^2 = 1$.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Data Set</th>
<th>AI</th>
<th>EM</th>
<th>PXEM</th>
<th>$\hat{\gamma}_{11}$</th>
<th>$\hat{\gamma}_{12}$</th>
<th>$\hat{\gamma}_{22}$</th>
<th>$\hat{\sigma}^2$</th>
<th>$\hat{\rho}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>6</td>
<td>73</td>
<td>27</td>
<td>0.88</td>
<td>0.17</td>
<td>1.56</td>
<td>0.84</td>
<td>0.15</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>6</td>
<td>119</td>
<td>31</td>
<td>0.88</td>
<td>-0.66</td>
<td>1.67</td>
<td>1.16</td>
<td>-0.54</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>5</td>
<td>434</td>
<td>57</td>
<td>0.50</td>
<td>0.29</td>
<td>0.93</td>
<td>1.26</td>
<td>0.43</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>6</td>
<td>56</td>
<td>23</td>
<td>1.44</td>
<td>-0.24</td>
<td>1.99</td>
<td>1.03</td>
<td>-0.14</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>6</td>
<td>65</td>
<td>24</td>
<td>1.95</td>
<td>0.36</td>
<td>0.65</td>
<td>1.01</td>
<td>0.32</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>6</td>
<td>138</td>
<td>26</td>
<td>0.77</td>
<td>0.31</td>
<td>0.83</td>
<td>0.86</td>
<td>0.38</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>10</td>
<td>&gt;500</td>
<td>90</td>
<td>0.17</td>
<td>0.20</td>
<td>0.90</td>
<td>0.84</td>
<td>0.49</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>7</td>
<td>&gt;500</td>
<td>78</td>
<td>0.22</td>
<td>-0.17</td>
<td>0.99</td>
<td>1.19</td>
<td>-0.36</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>fail</td>
<td>60</td>
<td>26</td>
<td>1.62</td>
<td>0.18</td>
<td>0.28</td>
<td>0.78</td>
<td>0.26</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>5</td>
<td>57</td>
<td>26</td>
<td>1.21</td>
<td>0.19</td>
<td>0.90</td>
<td>0.76</td>
<td>0.19</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>6</td>
<td>128</td>
<td>29</td>
<td>0.74</td>
<td>0.35</td>
<td>1.18</td>
<td>0.83</td>
<td>0.38</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>fail</td>
<td>279</td>
<td>45</td>
<td>0.73</td>
<td>0.34</td>
<td>0.51</td>
<td>0.94</td>
<td>0.56</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>7</td>
<td>65</td>
<td>24</td>
<td>2.46</td>
<td>0.57</td>
<td>0.64</td>
<td>0.99</td>
<td>0.46</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>5</td>
<td>294</td>
<td>47</td>
<td>0.52</td>
<td>0.30</td>
<td>1.25</td>
<td>1.12</td>
<td>0.37</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>6</td>
<td>84</td>
<td>24</td>
<td>1.31</td>
<td>0.41</td>
<td>0.85</td>
<td>0.96</td>
<td>0.38</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>6</td>
<td>46</td>
<td>23</td>
<td>2.80</td>
<td>0.52</td>
<td>1.08</td>
<td>0.98</td>
<td>0.30</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>9</td>
<td>166</td>
<td>34</td>
<td>0.90</td>
<td>0.32</td>
<td>0.53</td>
<td>0.91</td>
<td>0.47</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>7</td>
<td>257</td>
<td>43</td>
<td>0.71</td>
<td>0.37</td>
<td>0.66</td>
<td>0.95</td>
<td>0.54</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>fail</td>
<td>&gt;500</td>
<td>158</td>
<td>0.56</td>
<td>0.67</td>
<td>0.95</td>
<td>0.87</td>
<td>0.92</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>7</td>
<td>120</td>
<td>26</td>
<td>2.11</td>
<td>1.33</td>
<td>1.55</td>
<td>0.95</td>
<td>0.74</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>7</td>
<td>38</td>
<td>23</td>
<td>2.91</td>
<td>0.38</td>
<td>0.65</td>
<td>0.71</td>
<td>0.28</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>5</td>
<td>63</td>
<td>23</td>
<td>1.53</td>
<td>0.13</td>
<td>0.93</td>
<td>1.04</td>
<td>0.11</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>fail</td>
<td>214</td>
<td>40</td>
<td>1.27</td>
<td>0.46</td>
<td>0.45</td>
<td>1.06</td>
<td>0.60</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>fail</td>
<td>&gt;500</td>
<td>116</td>
<td>1.56</td>
<td>1.34</td>
<td>1.26</td>
<td>0.91</td>
<td>0.95</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>fail</td>
<td>&gt;500</td>
<td>108</td>
<td>2.13</td>
<td>1.41</td>
<td>1.02</td>
<td>0.95</td>
<td>0.95</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>8</td>
<td>210</td>
<td>35</td>
<td>0.41</td>
<td>0.09</td>
<td>0.53</td>
<td>0.77</td>
<td>0.19</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>fail</td>
<td>&gt;500</td>
<td>286</td>
<td>0.29</td>
<td>0.54</td>
<td>1.13</td>
<td>0.79</td>
<td>0.94</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>6</td>
<td>144</td>
<td>32</td>
<td>1.04</td>
<td>0.50</td>
<td>0.93</td>
<td>0.96</td>
<td>0.51</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>fail</td>
<td>218</td>
<td>38</td>
<td>1.85</td>
<td>0.66</td>
<td>0.42</td>
<td>0.85</td>
<td>0.75</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>7</td>
<td>303</td>
<td>47</td>
<td>1.97</td>
<td>1.06</td>
<td>0.90</td>
<td>1.16</td>
<td>0.80</td>
</tr>
</tbody>
</table>
Figure 9.11: Theoretical convergence rate and the iteration sequence of $r_e^{(m)}$ for the AI, EM and PXEM algorithms using starting values of $G_p^{(0)} = I_2$ and $\sigma^2(0) = 1$ for the random coefficient data sets simulated using values of $\gamma = (1, 0, 1)'$. 

Data Set 1

Data Set 2

Data Set 3

Data Set 4

Data Set 5

Data Set 6

Data Set 7

Data Set 8

Data Set 9

Data Set 10
Figure 9.12: Theoretical convergence rate and the iteration sequence of $r_{\varepsilon}^{(m)}$ for the AI, EM and PXEM algorithms using starting values of $G_p^{(0)} = I_2$ and $\sigma^2(0) = 1$ for the random coefficient data sets simulated using values of $\gamma = (1, 0.5, 1)'$. 
9.3. RANDOM COEFFICIENT DATA SETS

Figure 9.13: Theoretical convergence rate and the iteration sequence of $r_e^{(m)}$ for the AI, EM and PXEM algorithms using starting values of $G_p^{(0)} = I_2$ and $\sigma^2(0) = 1$ for the random coefficient data sets simulated using values of $\gamma = (1, 0.8, 1)'$.
Table 9.18: The theoretical and empirical convergence rate for the AI, EM and PXEM algorithms for the simulated random coefficient data sets, rounded to 2 decimal places, obtained using the uninformed starting values of $G^{(0)}_p = I_2$ and $\sigma^{(0)} = 1$. The number of iterations to convergence for each of the algorithms are also presented. Setting 1 refers to the data sets simulated using $\gamma' = (1, 0, 1)$, Setting 2 refers to the data sets simulated using $\gamma' = (1, 0.5, 1)$, and Setting 3 refers to the data sets simulated using $\gamma' = (1, 0.8, 1)$. For the data sets where the EM algorithm did not converge within 500 iterations, $r_e^{(500)}$ is presented in place of the empirical convergence rate.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Data Set</th>
<th>Algorithm</th>
<th>Theoretical</th>
<th>Empirical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>AI</td>
<td>EM</td>
<td>PXEM</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>6</td>
<td>73</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>6</td>
<td>119</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>5</td>
<td>434</td>
<td>57</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>6</td>
<td>56</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>6</td>
<td>65</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>6</td>
<td>138</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>10</td>
<td>&gt;500</td>
<td>90</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>7</td>
<td>&gt;500</td>
<td>78</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>fail</td>
<td>60</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>5</td>
<td>57</td>
<td>26</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>6</td>
<td>128</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>fail</td>
<td>279</td>
<td>45</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>7</td>
<td>65</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>5</td>
<td>294</td>
<td>47</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>6</td>
<td>84</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>6</td>
<td>46</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>9</td>
<td>166</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>7</td>
<td>257</td>
<td>43</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>fail</td>
<td>&gt;500</td>
<td>158</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>7</td>
<td>120</td>
<td>26</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>7</td>
<td>38</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>5</td>
<td>63</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>fail</td>
<td>214</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>fail</td>
<td>&gt;500</td>
<td>116</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>fail</td>
<td>&gt;500</td>
<td>108</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>8</td>
<td>210</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>fail</td>
<td>&gt;500</td>
<td>286</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>6</td>
<td>144</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>fail</td>
<td>218</td>
<td>38</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>7</td>
<td>303</td>
<td>47</td>
</tr>
</tbody>
</table>
for the EM algorithm ranged from 0.63 to 1.00 indicating slow convergence. The average convergence rate for the EM algorithm over the 10 data sets in each setting increased slightly as the correlation between the random intercepts and slopes increased; from 0.87 for \( \gamma_{12} = 0 \), to 0.89 for \( \gamma_{12} = 0.5 \), and then to 0.91 for \( \gamma_{12} = 0.8 \). The convergence rate for the PXEM algorithm ranged from 0.44 to 0.93 indicating slow convergence, but faster than the EM algorithm in all cases. The average convergence rate for the PXEM algorithm over the 10 data sets in each setting increased slightly as the correlation between the random intercepts and slopes increased; from 0.62 for \( \gamma_{12} = 0 \), to 0.64 for \( \gamma_{12} = 0.5 \), and then to 0.70 for \( \gamma_{12} = 0.8 \).

We will now look at using hybrid schemes to analyse the data sets for which the AI algorithm failed to converge, that is data set 9 from setting 1, data sets 2 and 9 from setting 2, and data sets 3, 4, 5, 7 and 9 from setting 3. We will start by looking at the optimal number of local iterations for the local EM/AI and local PXEM/AI schemes. The data sets were analysed using the local hybrid schemes with the AI criterion using from 1 to 10 internal local iterations. Table 9.19 presents the optimal number of internal local iterations for the local EM/AI and local PXEM/AI schemes chosen based on investigation of the sequence of iteration types and plots of the iteration sequences for each of the variance parameters.

Table 9.19: The optimal number of internal local iterations for the local EM/AI and local PXEM/AI schemes for each of the random coefficient simulated data sets for which the AI algorithm failed to converge. Setting 1 refers to the data sets simulated using \( \gamma' = (1, 0, 1) \), Setting 2 refers to the data sets simulated using \( \gamma' = (1, 0.5, 1) \), and Setting 3 refers to the data sets simulated using \( \gamma' = (1, 0.8, 1) \).

<table>
<thead>
<tr>
<th>Setting</th>
<th>Data Set</th>
<th>local EM</th>
<th>local PXEM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

For the local EM/AI scheme, the optimal number of internal local iterations ranged from 2 to 4. Recall that the optimal number of internal local EM iterations was 3 for the Orthodontic data and 1 for the Ultrafiltration data. Based on these results, we will use 3 internal local EM iterations for the local EM/AI scheme. For the local PXEM/AI scheme, the optimal number of internal local iterations was 1 for four of the six data sets, and 2 for the remaining two data sets. For the two data sets for which 2 was the optimal number of internal local PXEM iterations, if only one internal local PXEM iteration was done, the local PXEM/AI scheme still only required a single external local PXEM iteration.
However after 2 internal local iterations the AI algorithm converged in fewer iterations than after one internal local iteration. Based on these results, we will use 1 internal local PXEM iteration for the local PXEM/AI scheme, as we did for the analysis of the Orthodontic and Ultrafiltration data.

To select the cut-off for $\Delta^{(m)}$ for the update criterion, the data were analysed using the EM/AI scheme with cut-offs for $\Delta^{(m)}$ ranging from 0.1 to 1.5. The results are presented in Table 9.20. The smallest cut-off that invoked EM iterations only for the data sets for which the AI algorithm failed was 1.3 for the data simulated with $\gamma_{12} = 0$ and 0.9 for the data simulated with $\gamma_{12} = 0.5$. For the data sets simulated using $\gamma_{12} = 0.8$, EM iterations were still invoked with a cut-off of 1.5 for data set 6 for which the AI algorithm converged.

To make sure that EM iterations were invoked due to $\Delta^{(m)}$ being greater than the cut-off, rather than due to an AI iteration failing, we need to look at the iteration sequences for the various cut-offs. Table 9.21 shows the sequence of the iteration types for the EM/AI scheme with the various cut-offs for the update criterion for the data sets for which the AI algorithm does not converge. Note that ”EM” denotes an EM iterations invoked due to $\Delta^{(m)}$ being greater than the cut-off, and ”EM*” denotes an EM iterations invoked due to an AI iteration failing.

The results in Table 9.20 suggest that a cut-off of greater than 1.5 was needed for the update criterion to invoke EM iterations only for the data sets for which the AI algorithm failed. However the results in Table 9.21 show that a cut-off of 1.5 failed to invoke EM iterations due to $\Delta^{(m)}$ being greater than the cut-off. The EM iterations were invoked due to an AI iteration failing. Therefore the update criterion with a cut-off of 1.5 is not able to detect convergence problems before they occur.

Note that for data sets 4 and 5 in setting 3 ($\gamma_{12} = 0.8$), a cut-off between 0.6 and 1.0 invoked an EM iteration for the first iteration due to $\Delta^{(1)}$ being greater than the cut-off. On the second iteration, $\Delta^{(2)}$ was less than the cut-off and so an AI iteration was used. However the AI matrix was not positive definite on the third iteration and so the process was restarted with an EM iteration using the variance parameters estimates from the previous EM iteration. So with a cut-off between 0.6 and 1.0, the algorithm successfully identified the convergence problems with the AI algorithm and began with an EM iteration, but then switched to AI iterations too soon. So to successfully avoid convergence problems, we need to use a cut-off of 0.5.
9.3. RANDOM COEFFICIENT DATA SETS

Table 9.20: The results of the analyses of the simulated random coefficient data sets using the EM/AI scheme with the update criterion using various cut-offs for $\Delta^{(m)}$. A tick indicates that at least one EM iteration was used. A cross indicates that all iterations were AI iterations. Setting 1 refers to the data sets simulated using $\gamma' = (1, 0, 1)$, Setting 2 refers to the data sets simulated using $\gamma' = (1, 0.5, 1)$, and Setting 3 refers to the data sets simulated using $\gamma' = (1, 0.8, 1)$.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Data Set</th>
<th>AI</th>
<th>0.1</th>
<th>0.2</th>
<th>0.3</th>
<th>0.4</th>
<th>0.5</th>
<th>0.6</th>
<th>0.7</th>
<th>0.8</th>
<th>0.9</th>
<th>1</th>
<th>1.1</th>
<th>1.2</th>
<th>1.3</th>
<th>1.4</th>
<th>1.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>6</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>6</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>5</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>6</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>6</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>6</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>10</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>7</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>2</td>
<td>9</td>
<td>fail</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>5</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>fail</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>7</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>6</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>5</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>fail</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>fail</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>fail</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>fail</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>fail</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>fail</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>7</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
Table 9.21: The sequence of iteration types for the EM/AI scheme with the update criterion with various cut-offs for \( \Delta^{(m)} \) for the simulated random coefficient data sets for which the AI algorithm fails to converge. "EM" denotes an EM iteration that was invoked due to \( \Delta^{(m)} > \) cut-off, and "EM*" denotes an EM iteration that was invoked due to the AI updates being outside of the parameter space (i.e. the AI matrix was non-positive definite). Setting 1 refers to the data sets simulated using \( \gamma' = (1, 0, 1) \), Setting 2 refers to the data sets simulated using \( \gamma' = (1, 0.5, 1) \), and Setting 3 refers to the data sets simulated using \( \gamma' = (1, 0.8, 1) \).

<table>
<thead>
<tr>
<th>Setting</th>
<th>Data Set</th>
<th>Cut-off for ( \Delta^{(m)} )</th>
<th>Iteration Sequence</th>
<th>Number of Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9</td>
<td>0.1</td>
<td>4 EM, 4 AI</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2-0.3</td>
<td>2 EM, 5 AI</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4-1.1</td>
<td>EM, 6 AI</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2-1.5</td>
<td>AI, EM*, 6 AI</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0.1</td>
<td>13 EM, 4 AI</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>5 EM, 5 AI</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3-0.4</td>
<td>3 EM, 5 AI</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-1.0</td>
<td>EM, 6 AI</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.1-1.2</td>
<td>AI, 4 EM*, 3 EM, 7 AI</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.3-1.5</td>
<td>AI, 5 EM*, 2 EM, 7 AI</td>
<td>15</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>0.1</td>
<td>30 EM, 5 AI</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>11 EM, 5 AI</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>6 EM, 5 AI</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>4 EM, 6 AI</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>3 EM, 6 AI</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6-0.7</td>
<td>2 EM, 7 AI</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8-1.1</td>
<td>EM, 8 AI</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2</td>
<td>AI, 2 EM*, 3 EM, 7 AI</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.3-1.5</td>
<td>AI, 3 EM*, EM, EM*, 7 AI</td>
<td>13</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>0.1</td>
<td>13 EM, 5 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>3 EM, 6 AI</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3-0.5</td>
<td>2 EM, 7 AI</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6-1.0</td>
<td>EM, EM*, 7 AI</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.1-1.5</td>
<td>AI, EM*, 3 EM, 8 AI</td>
<td>13</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>0.1</td>
<td>6 EM, 5 AI</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>3 EM, 6 AI</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3-0.5</td>
<td>2 EM, 7 AI</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6-1.0</td>
<td>EM, EM*, 7 AI</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.1-1.5</td>
<td>EM*, 2 EM, EM*, 7 AI</td>
<td>12</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>0.1</td>
<td>34 EM, 5 AI</td>
<td>39</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>12 EM, 5 AI</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>7 EM, 5 AI</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>4 EM, 6 AI</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>3 EM, 6 AI</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6-0.8</td>
<td>2 EM, 7 AI</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9-1.2</td>
<td>EM, 8 AI</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.3</td>
<td>AI, 2 EM*, 2 EM, 8 AI</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.4</td>
<td>AI, 2 EM*, EM, EM*, 8 AI</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.5</td>
<td>AI, 4 EM*, 8 AI</td>
<td>13</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>0.1-0.2</td>
<td>4 EM, 5 AI</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3-0.4</td>
<td>2 EM, 6 AI</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-1.1</td>
<td>EM, 10 AI</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2-1.4</td>
<td>AI, EM*, EM, 7 AI</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.5</td>
<td>AI, 2 EM*, 7 AI</td>
<td>10</td>
</tr>
</tbody>
</table>
9.3. RANDOM COEFFICIENT DATA SETS

While a cut-off of 0.5 for $\Delta^{(m)}$ successfully invoked EM iterations for the 8 data sets for which the AI algorithm failed, it also invoked EM iterations for 9 other data sets for which the AI algorithm successfully converged. Table 9.22 presents the number of iterations to convergence for these data sets for the AI algorithm and the EM/AI scheme with the update criterion with a cut-off of 0.5. For some of the data sets the EM/AI scheme converged in 1-2 less iterations than the AI algorithm, for other data sets, the AI algorithm data set converged in 1 less iteration than the EM/AI scheme, and for other data sets the two schemes converged in the same number if iterations. Therefore, even though using a cut-off of 0.5 for the update criterion results in unnecessary EM iterations for these data sets, the total number of iterations to convergence was similar to that obtained using the AI algorithm. And so the EM/AI scheme was robust to poor starting values while still able to converge quickly.

Table 9.22: Number of iterations to convergence for the AI algorithm and the EM/AI scheme with the update criterion with a cut-off of 0.5 for $\Delta^{(m)}$, for the data sets for which the hybrid scheme invoked EM iterations although the AI algorithm successfully converged. Setting 1 refers to the data sets simulated using $\gamma' = (1, 0, 1)$, Setting 2 refers to the data sets simulated using $\gamma' = (1, 0.5, 1)$, and Setting 3 refers to the data sets simulated using $\gamma' = (1, 0.8, 1)$.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Data Set</th>
<th>AI</th>
<th>EM/AI</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6</td>
<td>6</td>
<td>6 $\ (1/5)$</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>10</td>
<td>9 $\ (3/6)$</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>7</td>
<td>8 $\ (2/6)$</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>10</td>
<td>8 $\ (1/7)$</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>7</td>
<td>7 $\ (1/6)$</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>9</td>
<td>7 $\ (1/6)$</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>7</td>
<td>7 $\ (1/6)$</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>7</td>
<td>7 $\ (1/6)$</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>8</td>
<td>7 $\ (2/5)$</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>7</td>
<td>8 $\ (1/7)$</td>
</tr>
</tbody>
</table>
To select the cut-off for $p_{\zeta}(m)$, the data were analysed using the EM/AI scheme with the score criterion with cut-offs for $p_{\zeta}(m)$ ranging from 0.0001 to 0.5. The results are presented in Table 9.23. For the data sets simulated using $\gamma_{12} = 0$, a cut-off of 0.0001 still invoked EM iterations for data set 7 for which the AI algorithm converged. This also happened for data set 7 of the data simulated using $\gamma_{12} = 0.5$. For the data simulated using $\gamma_{12} = 0.8$, the largest cut-off that invoked EM iterations only for the data sets for which the AI algorithm failed was 0.001.

To make sure that EM iterations were invoked due to $p_{\zeta}(m)$ being less than the cut-off, rather than due to an AI iteration failing, we need to look at the iteration sequences for the various cut-offs. Table 9.24 shows the sequence of the iteration types for the EM/AI scheme with the score criterion using various cut-offs for $p_{\zeta}(m)$ for the data sets for which the AI algorithm did not converge.

The results in Table 9.20 suggest that a cut-off of less than 0.0001 for $p_{\zeta}(m)$ was needed for the score criterion to invoke EM iterations only for the data sets for which the AI algorithm failed. However the results in Table 9.24 show that a cut-off of less than 0.5 failed to invoke EM iterations due to $p_{\zeta}(m)$ being less than the cut-off. The EM iterations were invoked due to an AI iteration failing. To detect convergence problems before they occur, a cut-off of at least 0.5 is needed for the score criterion.

While a cut-off of 0.5 for $p_{\zeta}(m)$ successfully invoked EM iterations for the 8 data sets for which the AI algorithm failed, it also invoked EM iterations for 12 other data sets for which the AI algorithm successfully converged. Table 9.25 presents the number of iterations to convergence for these data sets for the AI algorithm and the EM/AI scheme with the score criterion with a cut-off of 0.5 for $p_{\zeta}(m)$. The number of iterations to convergence for the two schemes were very similar for all data sets except for data set 8 in setting 2 ($\gamma_{12} = 0.5$), for which the EM/AI scheme failed to converge within 500 iterations.

For data set 8 in setting 2, the EM/AI scheme started with a successful AI iteration, but then $p_{\zeta}(2) < 0.5$ and so an EM iteration was used. From this point $p_{\zeta}(m)$ was less than 0.5 at each iteration, not allowing the scheme to switch back to AI iterations. Recall from Table 9.17, using the uninformed starting values, the EM algorithm converged in 257 iterations and the AI iteration converged in 7 iterations for this data set. Note that using the EM/AI scheme, the first EM iteration used the variance parameter estimates from the first AI iteration rather than the original starting values. Hence the variance parameter estimates from the initial AI iteration were further from the solution than the uninformed starting values, and the scheme was not able to converge within 500 iterations using EM iterations. We will see later that if the original starting values had been used for the first EM iteration, the EM/AI scheme converges quickly.
### 9.3. RANDOM COEFFICIENT DATA SETS

Table 9.23: The results of the analyses of the simulated random coefficient data sets using the EM/AI scheme with various cut-offs for the score criterion. A tick indicates that at least one EM iteration was used. A cross indicates that all iterations were AI iterations.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Data Set</th>
<th>AI</th>
<th>Cut-off for $p_{S(m)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.0001</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>6</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>6</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>5</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>6</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>6</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>6</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>10</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>7</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>fail</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>5</td>
<td>✓</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>6</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>fail</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>7</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>5</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>6</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>6</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>9</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>7</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>fail</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>7</td>
<td>✓</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>7</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>5</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>fail</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>fail</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>fail</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>8</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>fail</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>6</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>fail</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>7</td>
<td>✓</td>
</tr>
</tbody>
</table>
Table 9.24: The sequence of iteration types for the EM/AI scheme with various cut-offs for the score criterion for the simulated random coefficient data sets for which the AI algorithm fails to converge. "EM" denotes an EM iterations invoked due to $p_{S(m)}$ being less than the cut-off, and "EM*" denotes an EM iterations invoked due to AI updates being outside the parameter space (i.e. the AI matrix was non-positive definite).

<table>
<thead>
<tr>
<th>Setting</th>
<th>Data Set</th>
<th>Cut-off for $p_{S(m)}$</th>
<th>Iteration Sequence</th>
<th>Number of Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9</td>
<td>0.0001-0.001</td>
<td>AI, 4 EM, 4 AI</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.005-0.5</td>
<td>EM, 6 AI</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0.0001-0.01</td>
<td>AI, EM*, 4 EM, EM*, EM, 7 AI</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.05-0.1</td>
<td>AI, EM*, 6 EM, 7 AI</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>EM, 6 AI</td>
<td>7</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>0.0001-0.01</td>
<td>AI, EM*, 2 EM, 2 EM*, 7 AI</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.05-0.1</td>
<td>AI, EM*, 3 EM, EM*, 7 AI</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>2 EM, 7 AI</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>0.0001-0.005</td>
<td>AI, EM*, EM, EM*, EM, 7 AI</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.01-0.1</td>
<td>AI, EM*, 3 EM, 7 AI</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>EM, 6 AI</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>0.0001-0.001</td>
<td>AI, EM*, EM, 2 EM*, 8 AI</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.005</td>
<td>AI, EM*, 2 EM, EM*, 8 AI</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.01-0.1</td>
<td>EM, EM*, 7 AI</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>3 EM, 6 AI</td>
<td>9</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>0.0001-0.001</td>
<td>AI, EM*, EM, 2 EM*, 7 AI</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.005-0.1</td>
<td>EM, EM*, 7 AI</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>3 EM, 6 AI</td>
<td>9</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>0.0001-0.001</td>
<td>AI, EM*, EM, 3 EM*, 7 AI</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.005-0.05</td>
<td>AI, EM*, 2 EM, EM*, EM, 7 AI</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.1-0.5</td>
<td>2 EM, 7 AI</td>
<td>9</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>0.0001-0.005</td>
<td>AI, 2 EM*, 7 AI</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.01-0.5</td>
<td>2 EM, 6 AI</td>
<td>8</td>
</tr>
</tbody>
</table>
Table 9.25: Number of iterations to convergence for the AI algorithm and the EM/AI scheme with the score criterion with a cut-off of 0.5 for $p_{S(m)}$, for the data sets for which the hybrid scheme invoked EM iterations although the AI algorithm successfully converged.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Data Set</th>
<th>AI</th>
<th>EM/AI</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>6</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>6</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>6</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>10</td>
<td>8 (1/7)</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>7</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>6</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>9</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>7</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>7</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>7</td>
<td>7 (2/5)</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>8</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>7</td>
<td>8 (1/7)</td>
</tr>
</tbody>
</table>
We will now look at the results of the analyses of all of the simulated random coefficient data sets using all of the hybrid schemes, that is the EM/AI, PXEM/AI, local EM/AI and local PXEM/AI schemes, with the AI, update and score criteria. We will use 3 internal local EM iterations for the local EM/AI scheme and 1 internal local PXEM iteration for the local PXEM/AI scheme, and we will use an EM type iteration if $\Delta^{(m)} > 0.5$ for the update criterion and if $p_{S(m)} < 0.5$ for the score criterion. The results are presented in Table 9.26.

Recall that the hybrid schemes with the AI criterion only use an EM type update when an AI update produces estimates of the variance parameters that are outside the parameter space (i.e. the variance parameter estimates result in a non-positive definite AI matrix). The first time and AI iteration fails, an EM iteration is done using the initial starting values. If a subsequent AI iteration fails, an EM iteration is done using the estimates from the previous EM iteration. Hence the results of the hybrid schemes with the AI criterion provide us with the minimum number of EM iterations that are needed to obtain estimates that are close enough to the REML solution to allow the AI algorithm to converge.

The hybrid schemes with the update and score criteria are designed to try and detect convergence problems with AI before it fails by assessing how close the iteration process is to convergence. They do not use the initial starting values for the first EM iteration, unless it is invoked in the first iteration. This was done to allow us to determine if the update or score criteria were able to detect convergence problems before they occurred. The results of the hybrid schemes with the AI criterion have been included for comparison with the schemes with the update and score criteria. Ideally the final sequence of iteration types for the schemes with the update and score criteria should match that of the schemes with the AI criterion.

For the data sets for which the AI algorithm failed, the results for the hybrid schemes with the update and score criteria were equivalent to the results with the AI criterion, except for the EM/AI scheme for data set 9 in setting 3 for which the score criterion invokes 2 EM iterations compared to 1 EM iteration for the AI and update criteria. These results indicate that the hybrid schemes with the update and score criteria successfully detected convergence problems before they occurred and were able to invoke just enough EM iterations to bring the variance parameter estimates close enough to the REML solution to allow the AI algorithm to converge.

The update criterion invoked EM type iterations unnecessarily for 9 of the data sets, whereas the score criterion did so for 12 of the data sets. For all of the data sets for which the update criterion unnecessarily invokes EM type iterations, the total number of iterations to convergence was less than that of the AI algorithm in almost all cases. Hence the initial EM iterations resulted in variance parameter estimates closer to the solution than the initial AI iterations. For the data sets for which the score criterion unnecessarily invoked EM type iterations the total number of iterations to convergence was less than that of the AI algorithm in almost all cases, except for data set 8 in setting 2 and data set 1 in setting 3.

For data set 8 in setting 2, the AI algorithm converged in 7 iterations and then EM algorithm converged in 257 iterations. The EM/AI algorithm with the score criterion
Table 9.26: Number of iterations to convergence for the simulated random coefficient data sets using the various hybrid schemes with the different criteria; 3 internal local EM iterations were used for the local EM/AI scheme, 1 internal local PXEM iteration was used for the local PXEM/AI scheme, a cut-off of 0.5 was used for $\Delta^{(m)}$ for the update criterion, and a cut-off of 0.5 was used for $p_{s(m)}$ for the score criterion. A "*" indicates that the iteration sequence was not ordered EM type iterations followed by AI iterations, rather the scheme began with an AI iteration and then switched to EM type iterations and then back to AI iterations.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Data set</th>
<th>Hybrid scheme</th>
<th>AI criterion</th>
<th>Update criterion</th>
<th>Score criterion</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>EM/AI</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>EM/AI</td>
<td>6</td>
<td>6</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>6</td>
<td>5</td>
<td>5 (1/4)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>6</td>
<td>6</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>6</td>
<td>6</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>EM/AI</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>EM/AI</td>
<td>6</td>
<td>6</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>6</td>
<td>6</td>
<td>5 (1/4)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>6</td>
<td>6</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>6</td>
<td>6</td>
<td>5 (1/4)</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>EM/AI</td>
<td>6</td>
<td>6</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>6</td>
<td>6</td>
<td>5 (1/4)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>6</td>
<td>6</td>
<td>5 (1/4)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>6</td>
<td>6</td>
<td>5 (1/4)</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>EM/AI</td>
<td>6</td>
<td>6 (1/5)</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>6</td>
<td>5 (1/4)</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>6</td>
<td>6 (1/5)</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>6</td>
<td>6 (1/5)</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>EM/AI</td>
<td>10</td>
<td>9 (3/6)</td>
<td>8 (1/7)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>10</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>10</td>
<td>8 (2/6)</td>
<td>8 (1/7)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>10</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>EM/AI</td>
<td>7</td>
<td>8 (2/6)</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>7</td>
<td>6 (1/5)</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>7</td>
<td>6 (1/5)</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>7</td>
<td>6 (1/5)</td>
<td>7</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>EM/AI</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>EM/AI</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Setting</td>
<td>Data set</td>
<td>Hybrid scheme</td>
<td>AI criterion</td>
<td>Update criterion</td>
<td>Score criterion</td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
<td>------------------------</td>
<td>--------------</td>
<td>------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>EM/AI</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>EM/AI</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>EM/AI</td>
<td>7</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>7</td>
<td>5 (1/4)</td>
<td>5 (1/4)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>7</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>7</td>
<td>5 (1/4)</td>
<td>5 (1/4)</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>EM/AI</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>EM/AI</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>EM/AI</td>
<td>6</td>
<td>6</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>6</td>
<td>6</td>
<td>5 (1/4)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>6</td>
<td>6</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>6</td>
<td>6</td>
<td>5 (1/4)</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>EM/AI</td>
<td>9</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>9</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>9</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>9</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>EM/AI</td>
<td>7</td>
<td>7 (1/6)</td>
<td>&gt;500 (&gt;499,1)*</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>7</td>
<td>6 (1/5)</td>
<td>10 (4/6)*</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>7</td>
<td>6 (1/5)</td>
<td>&gt;500 (&gt;499,1)*</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>9</td>
<td>6 (1/5)</td>
<td>10 (2/8)*</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>EM/AI</td>
<td>9 (1/8)</td>
<td>9 (3/6)</td>
<td>9 (2/7)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>EM/AI</td>
<td>7</td>
<td>7</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>7</td>
<td>7</td>
<td>5 (1/4)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>7</td>
<td>7</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>7</td>
<td>7</td>
<td>5 (1/4)</td>
</tr>
</tbody>
</table>
### 9.3. RANDOM COEFFICIENT DATA SETS

<table>
<thead>
<tr>
<th>Setting</th>
<th>Data set</th>
<th>Hybrid scheme</th>
<th>AI criterion</th>
<th>Update criterion</th>
<th>Score criterion</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1</td>
<td>EM/AI</td>
<td>7</td>
<td>7 (1/6)</td>
<td>7 (2/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>7</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>7</td>
<td>7 (1/6)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>7</td>
<td>7 (1/6)</td>
<td>fail</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>EM/AI</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>EM/AI</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>5 (1/4)</td>
<td>5 (1/4)</td>
<td>5 (1/4)</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>EM/AI</td>
<td>9 (2/7)</td>
<td>9 (2/7)</td>
<td>9 (3/6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>EM/AI</td>
<td>9 (2/7)</td>
<td>9 (2/7)</td>
<td>9 (3/6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>EM/AI</td>
<td>8</td>
<td>7 (2/5)</td>
<td>7 (2/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>8</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>8</td>
<td>7 (2/5)</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>8</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>EM/AI</td>
<td>9 (1/8)</td>
<td>9 (3/6)</td>
<td>9 (2/7)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>9 (1/8)</td>
<td>9 (2/7)</td>
<td>9 (3/6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>EM/AI</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>EM/AI</td>
<td>11 (1/10)</td>
<td>11 (1/10)</td>
<td>8 (2/6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
<td>7 (1/6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>EM/AI</td>
<td>7</td>
<td>8 (1/7)</td>
<td>8 (1/7)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>7</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>7</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>7</td>
<td>6 (1/5)</td>
<td>6 (1/5)</td>
</tr>
</tbody>
</table>
started with an AI iteration but then switched to EM iterations due to $p_{s(m)} < 0.5$, failing to converge within 500 iterations. Note that $G^{(1)}$ from the AI iteration was not positive definite. The AI algorithm was able to recover from this and converge, however, the EM algorithm made very slow progress from this point. We will see later in this chapter that we are able to achieve fast convergence if we reset to the initial starting values when the first EM iteration is invoked. The local/EM AI scheme with the score criterion also started with an AI iteration, but then switched to local EM iterations due to $p_{s(m)} < 0.5$, failing to converge within 500 iterations. The PXEM/AI and local PXEM/AI schemes with the score criterion converged quickly, however their iteration sequence started with an AI iteration, then switched to PXEM or local PXEM iterations due to $p_{s(m)} < 0.5$, and then switched back to AI iterations. A "*" is used in Table 9.26 to indicate that the iteration sequence was not ordered EM type iterations and then AI iterations. Rather the schemes began with an AI iteration and then switched to EM type iterations and then back to AI iterations.

The iteration sequences for the local schemes with the score criterion for data set 1 in setting 3 consisted entirely of local iterations. The local iterations only updated the elements of $G_p$ and held $\sigma^2 = 1$ and hence did not converge to the REML estimates. Holding $\sigma^2 = 1$ resulted in the score function for $\sigma^2$ being large, resulting in a large value for the score statistic and hence $p_{s(m)}$ was less than 0.5 for all iterations and the algorithm is unable to switch to AI iterations. As discussed previously, this could possibly be resolved by adjusting $\sigma^2$ after $\sigma^2_u$ is updated at each iteration using Result A.19.

In summary, if we want a hybrid scheme that is able to detect convergence problems before they occur for a wide range of models and starting values, we need to accept that we will sometimes invoke EM type iterations in some cases where the AI algorithm converges successfully. By doing so, the scheme often converges in fewer iterations than the straight AI algorithm due to the initial EM iterations used by the hybrid schemes producing variance parameter estimates closer to the REML solution than the initial AI iterations using by the straight AI algorithm.

The results of the analyses of the simulated random coefficients data sets so far have been based on starting values of $\hat{\gamma} = (1, 0, 1)'$. We will now look at the results of the analyses using the AI, EM and PXEM algorithms with informed starting values obtained using the method described in Section 9.2.7.2; Henderson’s Method III was used to obtain an estimate of $\sigma^2$, and estimates of the elements of $\gamma$ were obtained by fitting a fixed intercept and slope for each individual and then calculating the variance-covariance matrix of the estimated coefficients. Table 9.27 contains the number of iterations for the AI, EM and PXEM algorithms using these start values. Table 9.28 contains the informed starting values and the REML estimates for each of the variance parameters.

For the data sets where the AI algorithm failed using the uninformed starting values, the AI algorithm converged using the informed starting values. However, note that the AI algorithm converged for data set 6 in setting 1 using the uninformed starting values but failed using the informed starting values. For the data sets for which the AI algorithm converged, the number of iterations was similar using the uninformed and informed starting values for each of the algorithms, with the algorithms with the informed starting values often taking a few more iterations to converge. This suggests that the method we
used to obtain starting values for random coefficient data sets was not optimal. This is
an area that requires further research.

Table 9.27: The number of iterations to convergence for the AI, EM and PXEM
algorithms for the simulated random coefficient data sets using uninformed and
informed starting values.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Data Set</th>
<th>Uninformed</th>
<th>Informed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>AI  EM  PXEM</td>
<td>AI  EM  PXEM</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>6  73  27</td>
<td>6  82  27</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>6  119 31</td>
<td>7  125 32</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>5  434 57</td>
<td>8  437 57</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>6  56  23</td>
<td>6  54  23</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>6  65  24</td>
<td>6  67  25</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>6  138 26</td>
<td>fail 142 34</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>10  &gt;500 90</td>
<td>11  &gt;500 91</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>7  &gt;500 78</td>
<td>9  &gt;500 78</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>fail 60 26</td>
<td>7  72  26</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>5  57  26</td>
<td>6  60  24</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>6  128 29</td>
<td>7  131 32</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>fail 279 45</td>
<td>7  281 46</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>7  65  24</td>
<td>7  69  25</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>5  294 47</td>
<td>7  297 47</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>6  84  24</td>
<td>7  93  28</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>6  46  23</td>
<td>7  43  21</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>9  166 34</td>
<td>7  169 37</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>7  257 43</td>
<td>7  259 44</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>fail  &gt;500 158</td>
<td>9  &gt;500 159</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>7  120 26</td>
<td>7  124 31</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>7  38  23</td>
<td>5  35  19</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>5  63  23</td>
<td>6  63  24</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>fail 214 40</td>
<td>7  217 41</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>fail  &gt;500 116</td>
<td>8  &gt;500 117</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>fail  &gt;500 108</td>
<td>9  &gt;500 109</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>8  210 35</td>
<td>7  212 41</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>fail  &gt;500 286</td>
<td>9  &gt;500 286</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>6  144 32</td>
<td>7  147 34</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>fail 218 38</td>
<td>7  219 41</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>7  303 47</td>
<td>7  306 47</td>
</tr>
</tbody>
</table>
Table 9.28: The informed starting values and the REML estimates for each of the simulated random coefficient data sets.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Data Set</th>
<th>$\gamma_{11}$</th>
<th>$\gamma_{12}$</th>
<th>$\gamma_{22}$</th>
<th>$\sigma^2$</th>
<th>$\gamma_{11}$</th>
<th>$\gamma_{12}$</th>
<th>$\gamma_{22}$</th>
<th>$\sigma^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1.76</td>
<td>-0.12</td>
<td>1.60</td>
<td>0.84</td>
<td>0.88</td>
<td>0.17</td>
<td>1.56</td>
<td>0.84</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2.07</td>
<td>-0.98</td>
<td>1.74</td>
<td>1.16</td>
<td>0.88</td>
<td>-0.66</td>
<td>1.67</td>
<td>1.16</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1.79</td>
<td>-0.09</td>
<td>1.07</td>
<td>1.26</td>
<td>0.50</td>
<td>0.29</td>
<td>0.93</td>
<td>1.26</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2.46</td>
<td>-0.51</td>
<td>1.98</td>
<td>1.03</td>
<td>1.44</td>
<td>-0.24</td>
<td>1.99</td>
<td>1.03</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>2.97</td>
<td>0.07</td>
<td>0.72</td>
<td>1.01</td>
<td>1.95</td>
<td>0.36</td>
<td>0.65</td>
<td>1.01</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>2.27</td>
<td>-0.09</td>
<td>0.90</td>
<td>0.86</td>
<td>0.77</td>
<td>0.31</td>
<td>0.83</td>
<td>0.86</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>1.10</td>
<td>-0.17</td>
<td>1.18</td>
<td>0.84</td>
<td>0.17</td>
<td>0.20</td>
<td>0.90</td>
<td>0.84</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>1.54</td>
<td>-0.48</td>
<td>1.06</td>
<td>1.19</td>
<td>0.22</td>
<td>-0.17</td>
<td>0.99</td>
<td>1.19</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>2.37</td>
<td>-0.11</td>
<td>0.47</td>
<td>0.78</td>
<td>1.62</td>
<td>0.18</td>
<td>0.28</td>
<td>0.78</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>1.96</td>
<td>0.01</td>
<td>1.02</td>
<td>0.76</td>
<td>1.21</td>
<td>0.19</td>
<td>0.90</td>
<td>0.76</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1.61</td>
<td>0.14</td>
<td>1.24</td>
<td>0.83</td>
<td>0.74</td>
<td>0.35</td>
<td>1.18</td>
<td>0.83</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1.68</td>
<td>0.05</td>
<td>0.65</td>
<td>0.94</td>
<td>0.73</td>
<td>0.34</td>
<td>0.51</td>
<td>0.94</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>3.52</td>
<td>0.11</td>
<td>0.84</td>
<td>0.99</td>
<td>2.46</td>
<td>0.57</td>
<td>0.64</td>
<td>0.99</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1.66</td>
<td>-0.02</td>
<td>1.33</td>
<td>1.12</td>
<td>0.52</td>
<td>0.30</td>
<td>1.25</td>
<td>1.12</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>2.31</td>
<td>0.04</td>
<td>0.98</td>
<td>0.96</td>
<td>1.31</td>
<td>0.41</td>
<td>0.85</td>
<td>0.96</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>4.43</td>
<td>-0.01</td>
<td>1.19</td>
<td>0.98</td>
<td>2.80</td>
<td>0.52</td>
<td>1.08</td>
<td>0.98</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>1.80</td>
<td>0.05</td>
<td>0.62</td>
<td>0.91</td>
<td>0.90</td>
<td>0.32</td>
<td>0.53</td>
<td>0.91</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>1.76</td>
<td>0.08</td>
<td>0.71</td>
<td>0.95</td>
<td>0.71</td>
<td>0.37</td>
<td>0.66</td>
<td>0.95</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>1.45</td>
<td>0.38</td>
<td>0.99</td>
<td>0.87</td>
<td>0.56</td>
<td>0.67</td>
<td>0.95</td>
<td>0.87</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>3.27</td>
<td>1.08</td>
<td>1.59</td>
<td>0.95</td>
<td>2.11</td>
<td>1.33</td>
<td>1.55</td>
<td>0.95</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>3.50</td>
<td>0.16</td>
<td>0.70</td>
<td>0.71</td>
<td>2.91</td>
<td>0.38</td>
<td>0.65</td>
<td>0.71</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2.58</td>
<td>-0.18</td>
<td>0.98</td>
<td>1.04</td>
<td>1.53</td>
<td>0.13</td>
<td>0.93</td>
<td>1.04</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>2.31</td>
<td>0.13</td>
<td>0.54</td>
<td>1.06</td>
<td>1.27</td>
<td>0.46</td>
<td>0.45</td>
<td>1.06</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2.46</td>
<td>1.04</td>
<td>1.31</td>
<td>0.91</td>
<td>1.56</td>
<td>1.34</td>
<td>1.26</td>
<td>0.91</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>3.18</td>
<td>1.11</td>
<td>1.07</td>
<td>0.95</td>
<td>2.13</td>
<td>1.41</td>
<td>1.02</td>
<td>0.95</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>1.22</td>
<td>-0.08</td>
<td>0.67</td>
<td>0.77</td>
<td>0.41</td>
<td>0.09</td>
<td>0.53</td>
<td>0.77</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>1.11</td>
<td>0.30</td>
<td>1.16</td>
<td>0.79</td>
<td>0.29</td>
<td>0.54</td>
<td>1.13</td>
<td>0.79</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>2.00</td>
<td>0.23</td>
<td>1.01</td>
<td>0.96</td>
<td>1.04</td>
<td>0.50</td>
<td>0.93</td>
<td>0.96</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>2.67</td>
<td>0.42</td>
<td>0.51</td>
<td>0.85</td>
<td>1.85</td>
<td>0.66</td>
<td>0.42</td>
<td>0.85</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>3.23</td>
<td>0.85</td>
<td>1.15</td>
<td>1.16</td>
<td>1.97</td>
<td>1.06</td>
<td>0.90</td>
<td>1.16</td>
</tr>
</tbody>
</table>
9.4 Multi-Environment Plant Variety Data

9.4.1 Simulated Multi-Environment Plant Variety Data

We will now investigate the performance of the various schemes for some simulated multi-environment plant variety trial data sets. Plant variety improvement programs involve the evaluation of potential new varieties in designed experiments at a number of trial locations and possibly over several years. These trials are known as multi-environment trials (METs) where an environment constitutes a particular location/year combination (Smith et al., 2001b). We consider a MET with 3 replicates of each of 100 varieties in each of 5 environments. The data are ordered replicates within varieties within environments. If $y$ is the $1500 \times 1$ vector of observations then the model is given by

$$y = X\tau + Zu + e$$

where $\tau$ is the $5 \times 1$ vector of fixed environment means and $X = I_5 \otimes 1_{100} \otimes 1_3$ is the $1500 \times 5$ associated design matrix. $u$ is the $500 \times 1$ vector of random effects for the 100 varieties in each of the 5 environments, and $Z$ is the $1500 \times 500$ associated design matrix. $e$ is the $900 \times 1$ vector of residuals.

We assume that the joint distribution of $u$ and $e$ is given by

$$\begin{bmatrix} u \\ e \end{bmatrix} \sim N\left( \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} G_5 \otimes I_{100} & 0 \\ 0 & \sigma^2 I_{1500} \end{bmatrix} \right)$$

By assuming that $G = G_5 \otimes I_{100}$, we are assuming independence between varieties. The symmetric positive definite matrix $G_5$ is often referred to as the genetic variance matrix. The diagonal elements represent genetic variances for individual environments and the off-diagonal elements represent genetic covariances between pairs of environments.

The data were simulated based on environment means $\tau' = (1.0, 0.5, 1.0, 0.4, 0.9)$ and

$$G_5 = \begin{bmatrix} 1.1 & 0.5 & 0.9 & 0.5 & 0.8 \\ 0.5 & 0.5 & 0.6 & 0.3 & 0.5 \\ 0.9 & 0.6 & 2.1 & 0.6 & 0.9 \\ 0.5 & 0.3 & 0.6 & 0.5 & 0.5 \\ 0.8 & 0.5 & 0.9 & 0.5 & 1.1 \end{bmatrix}$$

METs are commonly unbalanced with all varieties not necessarily replicated equally or present in all environments (Smith et al., 2005). Thirty data sets were simulated as described above and then a percentage of the observations were deleted; 40% of the observations in the first 10 data sets were deleted, 50% of the observations in the second 10 data sets were deleted and 60% of the observations in the last 10 data sets were deleted.

The number of iterations to convergence for the AI, EM and PXEM algorithms, using both uninformed and informed starting values, are given in Table 9.29. The uninformed starting values used were $G_5^{(0)} = I_5$ and $\sigma^2(0) = 1$, and the informed starting values were
Table 9.29: The number of iterations to convergence for the AI, EM and PXEM algorithms using uninformed and informed starting values for the simulated MET data sets. The rows in italics are the data sets for which the AI algorithm converged to a solution such that $G < 0$.

<table>
<thead>
<tr>
<th>% Missing</th>
<th>Data Set</th>
<th>Uninformed</th>
<th>Informed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>AI</td>
<td>EM</td>
</tr>
<tr>
<td>40</td>
<td>1</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>fail</td>
<td>398</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td>50</td>
<td>1</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td>60</td>
<td>1</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>fail</td>
<td>&gt;500</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>fail</td>
<td>355</td>
</tr>
</tbody>
</table>

obtained using Henderson’s Method III as described in Section 9.2.7.1. Note that the iteration sequence was terminated if the algorithm did not converge within 500 iterations, represented by " > 500" in Table 9.29.

Firstly, we will consider the results obtained using the uninformed starting values. The uninformed starting values of $G_p^{(0)} = I$ and $\sigma_2^{(0)} = 1$ were too far from the REML solution to enable the AI algorithm to converge for all of the data sets. The EM algorithm did not converge within 500 iterations for all but two of the data sets. The PXEM algorithm
converged in fewer than 500 iterations for 19 of the 30 data sets. For the two data sets where both the EM and PXEM algorithms converged in fewer than 500 iterations, the PXEM algorithm outperformed the EM algorithm, converging in 102 and 42 iterations, compared to 398 and 355 iterations, respectively. For three of the data sets for which the EM algorithm did not converge within 500 iterations, the PXEM algorithm clearly outperformed the EM algorithm converging in fewer than 100 iterations. However, for a number of the data sets, the EM and PXEM algorithms did not converge within 500 iterations and hence we are not able to compare their performance. The EM and PXEM algorithms are guaranteed to converge to a local maximum no matter what starting values are used, however we see here that they were very slow to converge using the uninformed starting values. Overall, the results indicate that none of the algorithms performed well using the uninformed starting values; the AI algorithm failed to stay within the parameter space (i.e. the variance parameter estimates resulted in a non-positive definite AI matrix), and the EM and PXEM algorithms converged very slowly.

Using the informed starting values, the AI algorithm converged in relatively few iterations for all data sets indicating that the starting values obtained using Henderson’s Method III were close enough to the solution to allow the AI algorithm to converge. The EM algorithm converged in fewer than 500 iterations using the uninformed starting values but took over 500 iterations using the informed starting values. The results for the PXEM algorithm with the informed starting values were very similar to the results based on the uninformed starting values. The PXEM algorithm converged in fewer than 500 iterations for the same 19 data sets, and did so in a very similar number of iterations. Therefore these results show that the performance of the AI algorithm was vastly improved through the use of the informed starting values calculated using Henderson’s Method III. Whereas the use of the informed starting values had very little effect on the performance of the EM and PXEM algorithms.

In summary, the AI algorithm clearly outperformed the EM and PXEM algorithms when informed starting values were used but failed to converge using the uninformed starting values of $G_p^{(0)} = I$ and $\sigma^2(0) = 1$. The use of informed starting values did not improve the performance of the EM and PXEM algorithms. The performance of the algorithms did not vary with the amount of missing data, other than the AI algorithm converging in slightly fewer iterations for the data sets with 60% missing compared to the data sets with 40% and 50% missing.

Recall from Section 7.2 that we chose to terminate the AI algorithm if the AI matrix became non-positive definite, and hence defined the parameter space as all values of the variance parameters that result in a positive definite AI matrix. This allowed the AI algorithm to converge in some cases where $G^{(m)}$ became non-positive definite or $\sigma^2(m)$ became negative during the iteration process. However, the allowable parameter space for the solution is all values of the variance parameters such that $G > 0$ and $\sigma^2 > 0$. For data sets 4 and 6 with 40% missing, data set 9 with 50% missing and data sets 6, 7 and 8 with 60% missing, the AI algorithm converged to variance parameter estimates such that $G$ was not positive definite ($G < 0$).

Thompson et al. (2003) state that in multivariate applications, it is difficult to ensure that
the REML estimates of the variance parameters for complex variance structures, such as the unstructured variance model used here, remain within the parameter space. Smith et al. (2001b) proposed a factor analytic (FA) variance structure as an alternative form for the genetic variance matrix. The FA model has been found to provide a good parsimonious approximation to the unstructured (US) form and is generally more computationally robust (Thompson et al., 2003). Kelly et al. (2007) showed through a series of simulations that the FA model was generally the preferred model over US, even for a number of data sets where the underlying variance structure was generated from a US model. The superiority of the FA models over US was particularly evident when the data involved a smaller number of varieties. We have simulated data sets based on 100 varieties and 5 environments. If we had increased the number of varieties, or reduced the number of environments, we may have had more success with the unstructured model.

For the data sets for which the AI algorithm converged to variance parameter estimates outside of the allowable parameter space for the solution, the EM and PXEM algorithms did not converge within 500 iterations. At the 500th iteration, the smallest eigenvalue of $G^{(500)}$ was close to or equal to zero for the PXEM algorithm, indicating that the estimates were heading towards the boundary. The smallest eigenvalue of $G^{(500)}$ for the EM algorithm was not as close to zero, as the algorithm was not as close to convergence as the PXEM algorithm at the 500th iteration. When the final variance parameter estimates obtained using the AI algorithm were used as starting values, the EM and PXEM algorithms converged to variance parameter estimates such that the smallest eigenvalue of $G^{(500)}$ was zero, hence $G^{(500)}$ was positive semi-definite ($G^{(500)} \succeq 0$). Hence, the EM and PXEM algorithms converged to variance parameters estimates on the boundary of the allowable parameter space for the solution, whereas the AI algorithm converged to variance parameters outside of the allowable parameter space for the solution. These results indicate that the REML solution was on the boundary and suggest that the unstructured model was over-parameterised and that a simpler variance model was needed for these data sets.

We will now look at the convergence rates for the AI, EM and PXEM algorithms. Figures 9.14, 9.15 and 9.16 display the iteration sequence of $r_e^{(m)}$ for the AI, EM and PXEM algorithms using the informed starting values for all of the simulated MET data sets. Table 9.30 presents the theoretical and empirical convergence rates for each of the algorithms. The empirical convergence rates are based on the iteration sequences of $r_e^{(m)}$ obtained using the informed starting values. For the data sets where the EM and PXEM algorithms did not converge within 500 iterations, $r_e^{(500)}$ is presented in place of the empirical convergence rate.

The plots and the results in the table indicate that $r_e^{(m)}$ for the AI algorithm did not stabilise for data sets 1, 4, 5 and 9 of the data sets with 40% missing, data set 1 of the data sets with 50% missing and data sets 1, 2, 3, 5, 7, 8, 9 and 10 of the data sets with 60% missing. Note that when these analyses were repeated with stricter convergence criteria, that is $\sqrt{\sum \left( \kappa_i^{(m+1)} - \kappa_i^{(m)} \right)^2 / \sum \kappa_i^{(m+1)}^2}$ less than $10^{-10}$ and $10^{-12}$, the empirical convergence rate stabilised for some of these data sets but not for others. The final variance parameter estimates for the analyses using $10^{-8}$, $10^{-10}$ and $10^{-12}$ were equal to at least eight decimal places, suggesting that the algorithm had converged in all cases. The reason for the instability of the empirical convergence rate for these data sets is
Table 9.30: The theoretical and empirical convergence rates for the AI, EM and PXEM algorithms for the simulated MET data sets, rounded to 2 decimal places. For the cases where the EM or PXEM algorithm did not converge within 500 iterations, $r_e^{(500)}$ is presented in place of the empirical convergence rate. The rows in italics are the data sets for which the AI algorithm converged to a solution such that $\hat{G} < 0$.

<table>
<thead>
<tr>
<th>% Missing</th>
<th>Data Set</th>
<th>Algorithm</th>
<th>Theoretical</th>
<th>Empirical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>AI</td>
<td>EM</td>
<td>PXEM</td>
</tr>
<tr>
<td>40</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>12 &gt;500</td>
<td>&gt;500</td>
<td>0.21</td>
<td>1.00</td>
</tr>
<tr>
<td>2</td>
<td>19 &gt;500</td>
<td>271</td>
<td>0.39</td>
<td>1.00</td>
</tr>
<tr>
<td>3</td>
<td>34 &gt;500</td>
<td>331</td>
<td>0.68</td>
<td>1.00</td>
</tr>
<tr>
<td>4</td>
<td>12 &gt;500</td>
<td>&gt;500</td>
<td>0.18</td>
<td>1.00</td>
</tr>
<tr>
<td>5</td>
<td>11 &gt;500</td>
<td>85</td>
<td>0.20</td>
<td>0.99</td>
</tr>
<tr>
<td>6</td>
<td>12 &gt;500</td>
<td>&gt;500</td>
<td>0.20</td>
<td>1.00</td>
</tr>
<tr>
<td>7</td>
<td>15 &gt;500</td>
<td>105</td>
<td>0.33</td>
<td>0.98</td>
</tr>
<tr>
<td>8</td>
<td>24 &gt;500</td>
<td>207</td>
<td>0.52</td>
<td>1.00</td>
</tr>
<tr>
<td>9</td>
<td>11 &gt;500</td>
<td>107</td>
<td>0.28</td>
<td>1.00</td>
</tr>
<tr>
<td>10</td>
<td>13 &gt;500</td>
<td>118</td>
<td>0.20</td>
<td>0.99</td>
</tr>
<tr>
<td>50</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>12 &gt;500</td>
<td>148</td>
<td>0.25</td>
<td>1.00</td>
</tr>
<tr>
<td>2</td>
<td>21 &gt;500</td>
<td>129</td>
<td>0.44</td>
<td>0.99</td>
</tr>
<tr>
<td>3</td>
<td>15 &gt;500</td>
<td>&gt;500</td>
<td>0.28</td>
<td>1.00</td>
</tr>
<tr>
<td>4</td>
<td>16 &gt;500</td>
<td>&gt;500</td>
<td>0.32</td>
<td>1.00</td>
</tr>
<tr>
<td>5</td>
<td>15 &gt;500</td>
<td>&gt;500</td>
<td>0.28</td>
<td>1.00</td>
</tr>
<tr>
<td>6</td>
<td>20 &gt;500</td>
<td>&gt;500</td>
<td>0.43</td>
<td>1.00</td>
</tr>
<tr>
<td>7</td>
<td>20 &gt;500</td>
<td>104</td>
<td>0.40</td>
<td>0.98</td>
</tr>
<tr>
<td>8</td>
<td>17 &gt;500</td>
<td>119</td>
<td>0.32</td>
<td>1.00</td>
</tr>
<tr>
<td>9</td>
<td>17 &gt;500</td>
<td>&gt;500</td>
<td>0.33</td>
<td>1.01</td>
</tr>
<tr>
<td>10</td>
<td>30 &gt;500</td>
<td>213</td>
<td>0.62</td>
<td>1.00</td>
</tr>
<tr>
<td>60</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>10 &gt;500</td>
<td>480</td>
<td>0.12</td>
<td>1.00</td>
</tr>
<tr>
<td>2</td>
<td>10 &gt;500</td>
<td>71</td>
<td>0.12</td>
<td>0.99</td>
</tr>
<tr>
<td>3</td>
<td>10 &gt;500</td>
<td>350</td>
<td>0.12</td>
<td>1.00</td>
</tr>
<tr>
<td>4</td>
<td>11 &gt;500</td>
<td>163</td>
<td>0.14</td>
<td>1.00</td>
</tr>
<tr>
<td>5</td>
<td>10 &gt;500</td>
<td>63</td>
<td>0.12</td>
<td>0.99</td>
</tr>
<tr>
<td>6</td>
<td>14 &gt;500</td>
<td>&gt;500</td>
<td>0.22</td>
<td>1.00</td>
</tr>
<tr>
<td>7</td>
<td>11 &gt;500</td>
<td>&gt;500</td>
<td>0.12</td>
<td>1.00</td>
</tr>
<tr>
<td>8</td>
<td>9 &gt;500</td>
<td>&gt;500</td>
<td>0.10</td>
<td>1.00</td>
</tr>
<tr>
<td>9</td>
<td>10 &gt;500</td>
<td>309</td>
<td>0.15</td>
<td>1.00</td>
</tr>
<tr>
<td>10</td>
<td>11 328</td>
<td>39</td>
<td>0.18</td>
<td>0.97</td>
</tr>
</tbody>
</table>
Figure 9.14: Theoretical convergence rate and the iteration sequence of $r_c^{(m)}$ for the AI, EM and PXEM algorithms for the MET data sets with 40% of the observations missing. For each algorithm, the dots represent $r_c^{(m)}$ at each iteration, and the horizontal line represents the theoretical convergence rate.
Figure 9.15: Theoretical convergence rate and the iteration sequence of $r_e^{(m)}$ for the AI, EM and PXEM algorithms for the MET data sets with 50% of the observations missing. For each algorithm, the dots represent $r_e^{(m)}$ at each iteration, and the horizontal line represents the theoretical convergence rate.
Figure 9.16: Theoretical convergence rate and the iteration sequence of $r_e^{(m)}$ for the AI, EM and PXEM algorithms for the MET data sets with 60% of the observations missing. For each algorithm, the dots represent $r_e^{(m)}$ at each iteration, and the horizontal line represents the theoretical convergence rate.
something that needs to be looked into.

For the EM algorithm, $r_e^{(m)}$ rapidly approached the theoretical convergence rate within the first 100 iterations but then slowly crept toward the theoretical convergence rate from this point. As previously discussed, the EM algorithm with the informed starting values only converged within 500 iterations for one of the data sets (data set 10 of the data sets with 60% missing). Looking at Table 9.30 we see that the convergence rate for the EM algorithm was lowest for this data set, with a convergence rate of 0.97. Most of the data sets had a theoretical convergence rate of 1.00 for the EM algorithm, supporting the very slow convergence that we observed. The theoretical convergence rate for the EM algorithm for data set 9 of the data sets with 50% missing was 1.01. Dempster et al. (1977) and McLachlan and Krishnan (1997) state that exceptions to the convergence of the EM algorithm to a local (if not global) maximum of the likelihood occur if the rate matrix has eigenvalues exceeding unity. They suggest that in this case, $\hat{\kappa}$ could be a saddle point of the likelihood. They state that if the rate matrix has an eigenvalue which is unity in a neighbourhood of $\hat{\kappa}$, this implies a ridge in the likelihood through $\hat{\kappa}$. Note that data set 9 of the data sets with 50% missing was one of the data sets for which the REML solution was on the boundary of the allowable parameter space.

For the PXEM algorithm, $r_e^{(m)}$ rapidly approached the theoretical convergence rate within the first 100 iterations and then slowly crept toward the theoretical convergence rate from this point. However, for three of the data sets (data sets 6, 9 and 6 from the settings with 40%, 50% and 60% missing respectively), $r_e^{(m)}$ became unstable in later iterations. The REML solution was on the boundary for each of these data sets, and the theoretical convergence rate for these data sets was greater than one. The theoretical convergence rate was also equal to one for the other 3 data sets for which the REML solution was on the boundary (data sets 4 from the setting with 40% missing and data sets 7 and 8 from the setting with 60% missing).

In summary, the convergence rate for the AI algorithm ranged between 0.10 and 0.52 indicating reasonably fast convergence which we observed when the starting values were within the neighbourhood of the solution. The convergence rate for the EM algorithm ranged from 0.97 to 1.01 indicating very slow convergence, which we observed with the EM algorithm taking more than 500 iterations to converge for most data sets. The convergence rate for the PXEM algorithm was generally smaller than that of the EM algorithm, ranging from 0.68 to 1.02, yet still much slower than the AI algorithm. For the data sets where the convergence rate for the PXEM algorithm was 0.99 or greater, the algorithm did not converge within 500 iterations.

For all of the data sets we see that $r_e^{(m)}$ for the EM and PXEM algorithms was lower than that of the AI algorithm over the first few iterations, indicating that the EM and PXEM algorithms were competitive in the early iterations. The plots of the iteration sequences of the AI, EM and PXEM algorithms are not presented due to the space required to show them for all 16 variance parameters for all 30 data sets for both sets of starting values. Inspection of the plots, for both the uninformed and informed starting values, showed that the EM and PXEM updates for each of the variance parameters approached the REML solution rapidly over the early iterations, the PXEM updates doing so faster than the EM iterations. However, both algorithms then took increasingly smaller steps
towards the solution, converging very slowly once in the neighbourhood of the solution. The AI algorithm failed to converge for all data sets using the uninformed starting values, but converged rapidly for all data sets using the informed starting values, indicating that the AI algorithm converges rapidly in the neighbourhood of the solution. These results support the use of a hybrid scheme that uses EM type iterations initially to obtain estimates of the variance parameters that are in the neighbourhood of the REML solution, and then switches to AI iterations to ensure rapid convergence.

We will now look at using hybrid schemes to analyse the simulated MET data sets using the uninformed starting values. The optimal number of local iterations for the local EM/AI and local PXEM/AI schemes were determined by studying the results of the analysis of these data sets using the local EM/AI and local PXEM/AI schemes with the AI criterion with from 1 to 10 internal local iterations. After examining the number of iterations to convergence and the plots of the iteration sequences for each of the variance parameters, we chose to use 3 internal local EM iterations for the local EM/AI scheme, and 2 internal local PXEM iterations for the local PXEM/AI scheme.

To select the cut-off for $\Delta^{(m)}$ for the hybrid schemes with the update criterion, we analysed the simulated MET data using the EM/AI scheme with cutoffs for $\Delta^{(m)}$ ranging from 0.1 to 1.5. Recall that the AI algorithm failed for all data sets, hence EM iterations were invoked for all data sets. To make sure that EM iterations were invoked due to $\Delta^{(m)}$ being greater than the cut-off, rather than due to an AI iteration failing, we need to look at the iteration sequences for the various cut-offs. Table 9.31 shows the sequence of the iteration types for the EM/AI scheme with the update criterion using the various cut-offs for $\Delta^{(m)}$ for all of the data sets.

Table 9.32 summarises the results presented in Table 9.31 by presenting the minimum cut-off for $\Delta^{(m)}$ that invoked EM iterations before an AI iteration fails for each of the data sets for each of the settings. The results suggest that a cut-off of no greater than 0.1 was needed to ensure that EM iterations were invoked due to $\Delta^{(m)}$ being greater than the cut-off rather than due to an AI iteration failing. Hence we will use an EM type iteration if $\Delta^{(m)} > 0.1$ for the hybrid schemes with the update criterion.
### Table 9.31: The sequence of iteration types for the EM/AI scheme with the update criterion using various cut-offs for $\Delta^{(m)}$ for the analysis of the simulated MET data sets using the uninformed starting values. “EM” denotes an EM iteration invoked due to $\Delta^{(m)} >$ cut-off, and ”EM*” denotes an EM iteration invoked due to the estimates from an AI update being outside the parameter space (i.e. the AI matrix was non-positive definite).

<table>
<thead>
<tr>
<th>Setting</th>
<th>Data Set</th>
<th>Cut-off for $\Delta^{(m)}$</th>
<th>Iteration Sequence</th>
<th>Number of Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>40%</td>
<td>1</td>
<td>0.1</td>
<td>13 EM, 14 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>6 EM, EM*, 20 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>4 EM, 3 EM*, 20 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>3 EM, 4 EM*, 20 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.6</td>
<td>2 EM, 5 EM*, 20 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7-0.8</td>
<td>EM, 6 EM*, 20 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9-1.0</td>
<td>AI, EM*, 5 EM, 5 EM*, 23 AI</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.1</td>
<td>AI, 4 EM*, 2 EM, 5 EM*, 23 AI</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2</td>
<td>AI, 5 EM*, EM, 5 EM*, 23 AI</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.3-1.5</td>
<td>AI, 11 EM*, 23 AI</td>
<td>35</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>0.1</td>
<td>10 EM, 18 AI</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>6 EM, 20 AI</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>4 EM, EM*, 22 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>3 EM, 2 EM*, 22 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.7</td>
<td>2 EM, 3 EM*, 22 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8-1.1</td>
<td>EM, 4 EM*, 20 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2</td>
<td>AI, EM*, 2 EM, 4 EM*, 21 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.3-1.5</td>
<td>AI, 7 EM*, 21 AI</td>
<td>29</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>0.1</td>
<td>12 EM, 36 AI</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>6 EM, 29 AI</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>4 EM, 38 AI</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4-0.5</td>
<td>3 EM, EM*, 38 AI</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6-0.7</td>
<td>2 EM, 2 EM*, 38 AI</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8-1.1</td>
<td>EM, 3 EM*, 38 AI</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2-1.5</td>
<td>AI, 4 EM*, 39 AI</td>
<td>44</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>0.1</td>
<td>13 EM, 11 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>7 EM, 13 AI</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>5 EM, 17 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>4 EM, EM*, 17 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.6</td>
<td>3 EM, 2 EM*, 17 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7-0.8</td>
<td>2 EM, 3 EM*, 17 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9-1.1</td>
<td>EM, 4 EM*, 17 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2-1.5</td>
<td>AI, 8 EM*, 14 AI</td>
<td>23</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>0.1</td>
<td>8 EM, 10 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>5 EM, 11 AI</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3-0.4</td>
<td>3 EM, 13 AI</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.7</td>
<td>2 EM, EM*, 13 AI</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8-1.1</td>
<td>EM, 2 EM*, 13 AI</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2</td>
<td>AI, 2 EM*, EM, 3 EM*, 14 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.3-1.5</td>
<td>AI, 6 EM*, 14 AI</td>
<td>21</td>
</tr>
</tbody>
</table>
### Table 9.1: Unstructured G Model

<table>
<thead>
<tr>
<th>Setting</th>
<th>Data Set</th>
<th>Cut-off for $\Delta^{(m)}$</th>
<th>Iteration Sequence</th>
<th>Number of Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>40%</td>
<td>6</td>
<td>0.1</td>
<td>12 EM, 12 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>7 EM, 14 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>5 EM, 18 AI</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>4 EM, EM*, 18 AI</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.6</td>
<td>3 EM, 2 EM*, 18 AI</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7-0.9</td>
<td>2 EM, 3 EM*, 18 AI</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.0-1.2</td>
<td>EM, 4 EM*, 18 AI</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.3-1.5</td>
<td>AI, 8 EM*, 14 AI</td>
<td>23</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>0.1</td>
<td>8 EM, 14 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>5 EM, 14 AI</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>4 EM, 14 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>3 EM, 15 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>2 EM, 18 AI</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6-0.9</td>
<td>EM, EM*, 18 AI</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.0</td>
<td>AI, EM*, 3 EM, 2 EM*, 17 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.1-1.2</td>
<td>AI, 2 EM*, 2 EM, 2 EM*, 17 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.3-1.5</td>
<td>AI, 6 EM*, 17 AI</td>
<td>24</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>0.1</td>
<td>8 EM, 22 AI</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>5 EM, 27 AI</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>4 EM, 30 AI</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>3 EM, EM*, 30 AI</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.7</td>
<td>2 EM, 2 EM*, 30 AI</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8-1.1</td>
<td>EM, 3 EM*, 30 AI</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2-1.5</td>
<td>AI, 2 EM*, 23 AI</td>
<td>26</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>0.1</td>
<td>9 EM, 12 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>5 EM, 13 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>4 EM, 13 AI</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>3 EM, 16 AI</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.7</td>
<td>2 EM, 2 EM*, 16 AI</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8-1.2</td>
<td>EM, 2 EM*, 16 AI</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.3-1.5</td>
<td>AI, 7 EM*, 13 AI</td>
<td>21</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>0.1</td>
<td>16 EM, 11 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>8 EM, 12 AI</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>5 EM, 13 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>4 EM, 14 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>3 EM, 15 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6-0.8</td>
<td>2 EM, EM*, 15 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9-1.1</td>
<td>EM, 2 EM*, 15 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2-1.3</td>
<td>AI, 3 EM*, EM, 3 EM*, 16 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.4-1.5</td>
<td>AI, 7 EM*, 16 AI</td>
<td>24</td>
</tr>
<tr>
<td>Setting</td>
<td>Data Set</td>
<td>Cut-off for $\Delta^{(m)}$</td>
<td>Iteration Sequence</td>
<td>Number of Iterations</td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
<td>-----------------------------</td>
<td>--------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>50%</td>
<td>1</td>
<td>0.1</td>
<td>19 EM, 12 AI</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>10 EM, 14 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>7 EM, 2 EM*, 15 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>5 EM, 4 EM*, 15 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>4 EM, 5 EM*, 15 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6</td>
<td>3 EM, 6 EM*, 15 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7</td>
<td>2 EM, 7 EM*, 15 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8-1.1</td>
<td>EM, 8 EM*, 15 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2</td>
<td>AI, 4 EM*, EM, 5 EM*, 17 AI</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.3-1.5</td>
<td>AI, 13 EM*, 17 AI</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.1</td>
<td>10 EM, 19 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>6 EM, 20 AI</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>4 EM, 23 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>3 EM, EM*, 23 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.6</td>
<td>2 EM, 2 EM*, 23 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7-1.0</td>
<td>EM, 3 EM*, 23 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.1</td>
<td>AI, 10 EM*, 3 EM, 5 EM*, 23 AI</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2</td>
<td>AI, 11 EM*, 2 EM, 5 EM*, 23 AI</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.3-1.5</td>
<td>AI, 18 EM*, 23 AI</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.1</td>
<td>13 EM, 14 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>7 EM, 17 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>5 EM, 2 EM*, 17 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>4 EM, 3 EM*, 17 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.6</td>
<td>3 EM, 4 EM*, 17 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7-0.8</td>
<td>2 EM, 5 EM*, 17 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9-1.1</td>
<td>EM, 6 EM*, 17 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2-1.5</td>
<td>AI, 8 EM*, 18 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.1</td>
<td>19 EM, 15 AI</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>10 EM, 18 AI</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>7 EM, 3 EM*, 18 AI</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>5 EM, 5 EM*, 18 AI</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>4 EM, 6 EM*, 18 AI</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6-0.7</td>
<td>3 EM, 7 EM*, 18 AI</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8-0.9</td>
<td>2 EM, 8 EM*, 18 AI</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.0-1.2</td>
<td>EM, 9 EM*, 18 AI</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.3</td>
<td>AI, 2 EM*, EM, 6 EM*, 18 AI</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.4-1.5</td>
<td>AI, 9 EM*, 18 AI</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0.1</td>
<td>13 EM, 14 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>7 EM, 17 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>5 EM, 2 EM*, 17 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>4 EM, 3 EM*, 17 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.6</td>
<td>3 EM, 4 EM*, 17 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7-0.8</td>
<td>2 EM, 5 EM*, 17 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9-1.1</td>
<td>EM, 6 EM*, 17 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2-1.5</td>
<td>AI, 8 EM*, 18 AI</td>
<td>27</td>
</tr>
<tr>
<td>Setting</td>
<td>Data Set</td>
<td>Cut-off for $\Delta^{(m)}$</td>
<td>Iteration Sequence</td>
<td>Number of Iterations</td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
<td>----------------------------</td>
<td>--------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>50%</td>
<td>6</td>
<td>0.1</td>
<td>15 EM, 15 AI</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>8 EM, 21 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>6 EM, EM*, 22 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>4 EM, 3 EM*, 22 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.6</td>
<td>3 EM, 4 EM*, 22 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7-0.8</td>
<td>2 EM, 5 EM*, 22 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9-1.1</td>
<td>EM, 6 EM*, 22 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2-1.5</td>
<td>AI, 11 EM*, 20 AI</td>
<td>32</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>0.1</td>
<td>10 EM, 17 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>6 EM, 16 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>4 EM, 19 AI</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>3 EM, 19 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.6</td>
<td>2 EM, 20 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7-0.9</td>
<td>EM, EM*, 20 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.0-1.1</td>
<td>AI, EM*, 5 EM, 3 EM*, 20 AI</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2-1.3</td>
<td>AI, 3 EM*, 2 EM, 4 EM*, 10 AI</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.4-1.5</td>
<td>AI, 9 EM*, 20 AI</td>
<td>30</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>0.1</td>
<td>18 EM, 13 AI</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>10 EM, 16 AI</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>7 EM, 17 AI</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>5 EM, 3 EM*, 17 AI</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>3 EM, 5 EM*, 17 AI</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6-0.7</td>
<td>2 EM, 6 EM*, 17 AI</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8-1.0</td>
<td>EM, 7 EM*, 17 AI</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.1-1.2</td>
<td>AI, EM*, 3 EM, 9 EM*, 16 AI</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.3</td>
<td>AI, 2 EM*, EM, 10 EM*, 16 AI</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.4-1.5</td>
<td>AI, 13 EM*, 16 AI</td>
<td>30</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>0.1</td>
<td>22 EM, 15 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>12 EM, 17 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>8 EM, 23 AI</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>6 EM, 2 EM, 23 AI</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>4 EM, 4 EM*, 23 AI</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6</td>
<td>3 EM, 5 EM*, 23 AI</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7-0.8</td>
<td>2 EM, 6 EM*, 23 AI</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9-1.0</td>
<td>EM, 7 EM*, 23 AI</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.1</td>
<td>AI, EM*, 3 EM, 8 EM*, 21 AI</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2</td>
<td>AI, 2 EM*, 2 EM, 8 EM*, 21 AI</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.3</td>
<td>AI, 12 EM*, 21 AI</td>
<td>34</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>0.1</td>
<td>10 EM, 22 AI</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>6 EM, 19 AI</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>4 EM, 30 AI</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>3 EM, EM*, 30 AI</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.6</td>
<td>2 EM, 2 EM*, 30 AI</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7-1.0</td>
<td>EM, 3 EM*, 30 AI</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.1</td>
<td>AI, EM*, 3 EM, 4 EM*, 34 AI</td>
<td>43</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2</td>
<td>AI, 2 EM*, 2 EM, 4 EM*, 34 AI</td>
<td>43</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.3</td>
<td>AI, 3 EM*, EM, 4 EM*, 34 AI</td>
<td>43</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.4-1.5</td>
<td>AI, 8 EM*, 34 AI</td>
<td>43</td>
</tr>
<tr>
<td>Setting</td>
<td>Data Set</td>
<td>Cut-off for $\Delta^{(m)}$</td>
<td>Iteration Sequence</td>
<td>Number of Iterations</td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
<td>-----------------------------</td>
<td>--------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>60%</td>
<td>1</td>
<td>0.1</td>
<td>16 EM, 9 AI</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>9 EM, 12 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>6 EM, 2 EM*, 14 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>5 EM, 3 EM*, 14 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>4 EM, 4 EM*, 14 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6</td>
<td>3 EM*, 5 EM*, 14 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7-0.8</td>
<td>2 EM, 6 EM*, 14 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9-1.1</td>
<td>EM, 7 EM*, 14 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2</td>
<td>AI, EM*, 3 EM, 9 EM*, 14 AI</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.3</td>
<td>AI, EM*, 2 EM, 10 EM*, 14 AI</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.4-1.5</td>
<td>AI, 13 EM*, 14 AI</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.1</td>
<td>18 EM, 9 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>9 EM, 10 AI</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>6 EM, 12 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>4 EM, EM*, 13 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>3 EM, 2 EM*, 13 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6-0.7</td>
<td>2 EM, 3 EM*, 13 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8-0.9</td>
<td>EM, 4 EM*, 13 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.1</td>
<td>AI, EM*, 7 EM, 5 EM*, 15 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2</td>
<td>AI, 3 EM*, 5 EM, 5 EM*, 15 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.3</td>
<td>AI, 4 EM*, 3 EM, 6 EM*, 15 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.4</td>
<td>AI, 5 EM*, EM, 7 EM*, 15 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.5</td>
<td>AI, 13 EM*, 15 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.1</td>
<td>17 EM, 9 AI</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>9 EM, 12 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>6 EM, 2 EM*, 15 AI</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>4 EM, 4 EM*, 15 AI</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>3 EM, 5 EM*, 15 AI</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6-0.7</td>
<td>2 EM, 6 EM*, 15 AI</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8-0.9</td>
<td>EM, 7 EM*, 15 AI</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.0</td>
<td>AI, EM*, 6 EM, 11 EM*, 12 AI</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.1</td>
<td>AI, 2 EM*, 4 EM, 12 EM*, 12 AI</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2</td>
<td>AI, 3 EM*, 3 EM, 12 EM*, 12 AI</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.3-1.5</td>
<td>AI, 18 EM*, 12 AI</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.1</td>
<td>17 EM, 11 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>9 EM, 12 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>6 EM, 16 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>5 EM, EM*, 16 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>4 EM, 2 EM*, 16 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6</td>
<td>3 EM, 3 EM*, 16 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7-0.8</td>
<td>2 EM, 4 EM*, 16 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9-1.1</td>
<td>EM, 5 EM*, 16 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2</td>
<td>AI, EM*, EM, 7 EM*, 14 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.3-1.5</td>
<td>AI, 9 EM*, 14 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0.1</td>
<td>15 EM, 10 AI</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>9 EM, 11 AI</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>6 EM, 12 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>4 EM, EM*, 13 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>3 EM, 2 EM*, 13 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6-0.7</td>
<td>2 EM, 3 EM*, 13 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8-1.0</td>
<td>EM, 4 EM*, 13 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.1-1.2</td>
<td>AI, EM*, EM, 5 EM*, 14 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.3-1.5</td>
<td>AI, 7 EM*, 14 AI</td>
<td>22</td>
</tr>
<tr>
<td>Setting</td>
<td>Data Set</td>
<td>Cut-off for $\Delta^{(m)}$</td>
<td>Iteration Sequence</td>
<td>Number of Iterations</td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
<td>-----------------------------</td>
<td>--------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>60%</td>
<td>6</td>
<td>0.1</td>
<td>20 EM, 11 AI</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>11 EM, 13 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>7 EM, 2 EM*, 15 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>5 EM, 4 EM*, 15 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>4 EM, 5 EM*, 15 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6</td>
<td>3 EM, 6 EM*, 15 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7-0.8</td>
<td>2 EM, 7 EM*, 15 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9-1.0</td>
<td>EM, 8 EM*, 15 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.1</td>
<td>AI, 2 EM*, 4 EM, 11 EM*, 16 AI</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2-1.5</td>
<td>AI, 17 EM, 16 AI</td>
<td>34</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>0.1</td>
<td>16 EM, 11 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>9 EM, EM*, 17 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>6 EM, 4 EM, 17 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>4 EM, 6 EM*, 17 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>3 EM, 7 EM*, 17 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6</td>
<td>2 EM, 9 EM*, 17 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7-0.9</td>
<td>EM, 9 EM*, 17 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>Al, EM*, 7 EM, 12 EM*, 15 AI</td>
<td>36</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.1-1.2</td>
<td>AI, 2 EM*, 6 EM, 12 EM*, 15 AI</td>
<td>36</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.3</td>
<td>Al, 3 EM*, 4 EM, 13 EM*, 15 AI</td>
<td>36</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.4</td>
<td>Al, 4 EM*, 3 EM, 13 EM, 15 AI</td>
<td>36</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.5</td>
<td>Al, 4 EM*, 2 EM, 14 EM*, 15 AI</td>
<td>36</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>0.1</td>
<td>18 EM, 9 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>10 EM, 13 AI</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>7 EM, 3 EM*, 13 AI</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>5 EM, 5 EM*, 13 AI</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>4 EM, 6 EM*, 13 AI</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6</td>
<td>3 EM, 7 EM*, 13 AI</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7</td>
<td>2 EM, 8 EM* 13 AI</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>0.8-1.0</td>
<td>EM, 9 EM*, 13 AI</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.1</td>
<td>AI, 4 EM*, 4 EM, 11 EM*, 14 AI</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.2-1.3</td>
<td>AI, 5 EM*, 3 EM, 11 EM*, 14 AI</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.4-1.5</td>
<td>Al, 19 EM* 14 AI</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>0.1</td>
<td>21 EM, 10 AI</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>11 EM, 10 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>8 EM, 13 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>6 EM, 2 EM*, 13 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>5 EM, 3 EM*, 13 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6</td>
<td>4 EM, 4 EM*, 13 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7</td>
<td>3 EM, 5 EM*, 13 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>0.8-0.9</td>
<td>2 EM, 6 EM*, 13 AI</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.0-1.2</td>
<td>EM, 7 EM*, 13 AI</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.3-1.5</td>
<td>Al, 11 EM*, 13 AI</td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>0.1</td>
<td>13 EM, 10 AI</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td>8 EM, 11 AI</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>5 EM, 11 AI</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.4</td>
<td>4 EM, 11 AI</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>3 EM, 13 AI</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>0.6-0.7</td>
<td>2 EM, EM*, 13 AI</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.8-0.9</td>
<td>EM, 2 EM*, 13 AI</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.0-1.1</td>
<td>AI, EM*, 4 EM, 5 EM*, 12 AI</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.2</td>
<td>Al, 2 EM*, 3 EM, 5 EM*, 12 AI</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.3</td>
<td>Al, 2 EM*, 2 EM, 6 EM*, 12 AI</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.4</td>
<td>Al, 3 EM*, EM, 6 EM*, 12 AI</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.5</td>
<td>Al, 10 EM*, 12 AI</td>
<td>23</td>
<td></td>
</tr>
</tbody>
</table>
Table 9.32: The minimum cut-off for $\Delta^{(m)}$ that invoked EM iterations before an AI iteration failed for each of the data sets for each of the settings for the analysis of the simulated MET data sets using the EM/AI scheme with the update criterion.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>40%</th>
<th>50%</th>
<th>60%</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.1</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>2</td>
<td>0.2</td>
<td>0.3</td>
<td>0.3</td>
</tr>
<tr>
<td>3</td>
<td>0.3</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>4</td>
<td>0.3</td>
<td>0.2</td>
<td>0.3</td>
</tr>
<tr>
<td>5</td>
<td>0.4</td>
<td>0.2</td>
<td>0.3</td>
</tr>
<tr>
<td>6</td>
<td>0.3</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>7</td>
<td>0.5</td>
<td>0.6</td>
<td>0.1</td>
</tr>
<tr>
<td>8</td>
<td>0.3</td>
<td>0.3</td>
<td>0.2</td>
</tr>
<tr>
<td>9</td>
<td>0.4</td>
<td>0.3</td>
<td>0.3</td>
</tr>
<tr>
<td>10</td>
<td>0.5</td>
<td>0.3</td>
<td>0.5</td>
</tr>
</tbody>
</table>

To select the cut-off for $p_{S(m)}$ for the hybrid schemes with the score criterion, the data were analysed using the EM/AI scheme with cut-offs for $p_{S(m)}$ ranging from 0.0001 to 0.9 using the uninformed starting values. For the variance component and random coefficient data sets we used cut-offs ranging from 0.0001 to 0.5. However, for some of the simulated MET data sets, a cut-off greater than 0.5 was required to invoke EM iterations before an AI iteration failed. Table 9.33 presents the sequence of iteration types for each of the data sets using the different cut-offs for $p_{S(m)}$.

Table 9.34 summarises the results presented in Table 9.33 by presenting the maximum cut-off for $p_{S(m)}$ that invoked EM iterations before an AI iteration failed for each of the data sets for each of the settings. The results suggest that a cut-off of greater than 0.9 was needed to ensure that EM iterations were invoked due to the score criterion rather than due to an AI iteration failing, since a cut-off of 0.9 failed to detect the convergence problems for data set 8 of the data sets with 50% missing before they occur. We analysed this data sets using cut-offs greater than 0.9 and determined that a cut-off of at least 0.95 was required for the score criterion to work. Hence we will use a cut-off of 0.95 for $p_{S(m)}$ for the hybrid schemes with the score criterion, that is, we will use an EM type iteration if $p_{S(m)} < 0.95$. 
Table 9.33: The sequence of iteration types for the EM/AI scheme with the score criterion using various cut-offs for $p_{S(m)}$ for the analysis of the simulated MET data sets using the uninformed starting values. "EM" denotes an EM iteration invoked due to $\Delta^{(m)} >$ cut-off, and "EM*" denotes an EM iteration invoked due to the estimates from an AI update being outside the parameter space (i.e. the AI matrix was non-positive definite).

<table>
<thead>
<tr>
<th>Setting</th>
<th>Data Set</th>
<th>Cut-off for $p_{S(m)}$</th>
<th>Iteration Sequence</th>
<th>Number of Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>40%</td>
<td>1</td>
<td>0.0001-0.01</td>
<td>2 EM, 5 EM*, EM, 16 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.05-0.1</td>
<td>3 EM, 4 EM*, EM, 16 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>4 EM, 3 EM*, 2 EM, 13 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6-0.7</td>
<td>5 EM, 2 EM*, 2 EM, 13 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8</td>
<td>6 EM, EM*, 3 EM, 16 AI</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9</td>
<td>10 EM, 16 AI</td>
<td>26</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>0.0001-0.001</td>
<td>2 EM, EM*, EM*, EM, 20 AI</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.005-0.1</td>
<td>4 EM, EM*, EM, 20 AI</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.9</td>
<td>7 EM, 19 AI</td>
<td>26</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>0.0001-0.001</td>
<td>2 EM, 2 EM*, EM, 35 AI</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.005-0.05</td>
<td>3 EM, EM*, EM, 35 AI</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.1</td>
<td>5 EM, 35 AI</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.7</td>
<td>6 EM, 29 AI</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8</td>
<td>7 EM, 34 AI</td>
<td>41</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9</td>
<td>8 EM, 35 AI</td>
<td>43</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>0.0001</td>
<td>3 EM, 2 EM*, 2 EM, 13 AI</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.0005-0.01</td>
<td>4 EM, EM*, 2 EM, 13 AI</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.05-0.1</td>
<td>7 EM, 13 AI</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.6</td>
<td>8 EM, 12 AI</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7</td>
<td>9 EM, 12 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8</td>
<td>10 EM, 12 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9</td>
<td>11 EM, 12 AI</td>
<td>23</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>0.0001-0.005</td>
<td>2 EM, EM*, EM, 12 AI</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.01-0.6</td>
<td>4 EM, 12 AI</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7-0.8</td>
<td>5 EM, 11 AI</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9</td>
<td>6 EM, 11 AI</td>
<td>16</td>
</tr>
</tbody>
</table>
### 9.4. MULTI-ENVIRONMENT PLANT VARIETY DATA

<table>
<thead>
<tr>
<th>Setting</th>
<th>Data Set</th>
<th>Cut-off for ( p_{\tilde{z}(m)} )</th>
<th>Iteration Sequence</th>
<th>Number of Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>40%</td>
<td>6</td>
<td>0.0001-0.001</td>
<td>3 EM, 2 EM*, EM, 15 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.005-0.01</td>
<td>4 EM, EM*, 2 EM, 14 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.05-0.1</td>
<td>7 EM, 14 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.6</td>
<td>8 EM, 13 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7-0.8</td>
<td>9 EM, 13 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9</td>
<td>10 EM, 13 AI</td>
<td>23</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>0.0001-0.1</td>
<td>3 EM, 15 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.7</td>
<td>4 EM, 14 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8</td>
<td>5 EM, 14 AI</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9</td>
<td>6 EM, 14 AI</td>
<td>20</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>0.0001</td>
<td>2 EM, 2 EM*, EM, 27 AI</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.0005-0.01</td>
<td>3 EM, EM*, EM, 27 AI</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.05-0.1</td>
<td>5 EM, 27 AI</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.8</td>
<td>6 EM, 25 AI</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9</td>
<td>7 EM, 24 AI</td>
<td>31</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>0.0001-0.001</td>
<td>2 EM, EM*, EM, 13 AI</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.005</td>
<td>4 EM, 13 AI</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.01-0.6</td>
<td>5 EM, 13 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7-0.8</td>
<td>6 EM, 12 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9</td>
<td>7 EM, 13 AI</td>
<td>20</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>0.0001-0.0005</td>
<td>2 EM, EM*, 15 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.001-0.01</td>
<td>2 EM, EM*, EM, 14 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.05-0.5</td>
<td>4 EM, 14 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6-0.8</td>
<td>5 EM, 13 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9</td>
<td>6 EM, 13 AI</td>
<td>19</td>
</tr>
</tbody>
</table>
## UNSTRUCTURED G MODEL

### Data Cut-off Iteration

<table>
<thead>
<tr>
<th>Setting</th>
<th>Data Set</th>
<th>Cut-off for $p_{G(m)}$</th>
<th>Iteration Sequence</th>
<th>Number of Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>50%</td>
<td>1</td>
<td>0.0001-0.001</td>
<td>2 EM, 7 EM*, EM, 14 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.005-0.01</td>
<td>3 EM, 6 EM*, EM, 14 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.05-0.1</td>
<td>4 EM, 5 EM*, EM, 14 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>6 EM, 3 EM*, 2 EM, 13 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6-0.7</td>
<td>7 EM, EM*, 2 EM, 13 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8</td>
<td>8 EM, EM*, 3 EM, 13 AI</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9</td>
<td>12 EM, 13 AI</td>
<td>25</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>0.0001</td>
<td>EM, 3 EM*, 23 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.0005-0.001</td>
<td>2 EM, 2 EM*, 23 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.005-0.01</td>
<td>2 EM, 2 EM*, EM, 21 AI</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.05-0.1</td>
<td>3 EM, EM*, EM, 21 AI</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.7</td>
<td>5 EM, 21 AI</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8-0.9</td>
<td>6 EM, 20 AI</td>
<td>26</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>0.0001-0.001</td>
<td>3 EM, 4 EM*, EM, 16 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.005-0.05</td>
<td>4 EM, 3 EM*, EM, 16 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.1</td>
<td>5 EM, 2 EM*, EM, 16 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.6</td>
<td>9 EM, 15 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7-0.9</td>
<td>10 EM, 15 AI</td>
<td>25</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>0.0001</td>
<td>2 EM, 8 EM*, EM, 17 AI</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.0005-0.005</td>
<td>3 EM, 7 EM*, EM, 17 AI</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.01-0.05</td>
<td>4 EM, 6 EM*, EM, 17 AI</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.1</td>
<td>5 EM, 5 EM*, 2 EM, 17 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.6</td>
<td>7 EM, 3 EM*, 3 EM, 16 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7</td>
<td>8 EM, 2 EM*, 3 EM, 16 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8</td>
<td>9 EM, EM*, 2 EM, 16 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9</td>
<td>14 EM, 16 AI</td>
<td>30</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>0.0001-0.001</td>
<td>3 EM, 4 EM*, EM, 16 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.005-0.05</td>
<td>4 EM, 3 EM*, EM, 16 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.1</td>
<td>5 EM, 2 EM*, EM, 16 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.6</td>
<td>9 EM, 15 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7-0.9</td>
<td>10 EM, 15 AI</td>
<td>25</td>
</tr>
<tr>
<td>Setting</td>
<td>Data Set</td>
<td>Cut-off for $p_{S(m)}$</td>
<td>Iteration Sequence</td>
<td>Number of Iterations</td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
<td>-------------------------</td>
<td>-------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>50%</td>
<td>6</td>
<td>0.0001-0.005</td>
<td>3 EM, 4 EM*, EM, 21 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.01-0.05</td>
<td>4 EM, 3 EM*, EM, 21 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.1</td>
<td>5 EM, 2 EM*, EM, 21 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.8</td>
<td>9 EM, 20 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9</td>
<td>10 EM, 19 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>0.0001-0.001</td>
<td>EM, EM*, EM, 19 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.005-0.1</td>
<td>3 EM, 19 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.8</td>
<td>4 EM, 19 AI</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9</td>
<td>5 EM, 18 AI</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>0.0001-0.001</td>
<td>EM, 7 EM*, EM, 17 AI</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.005-0.05</td>
<td>2 EM, EM*, EM, 4 EM*, EM, 17 AI</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.1</td>
<td>4 EM, 4 EM*, EM, 17 AI</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>4 EM, 4 EM*, 2 EM, 16 AI</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6-0.7</td>
<td>5 EM, 3 EM*, 2 EM, 16 AI</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8</td>
<td>6 EM, 2 EM*, 3 EM, 16 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9</td>
<td>7 EM, EM*, 3 EM, 16 AI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>0.0001</td>
<td>EM, 7 EM*, EM, 19 AI</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.0005</td>
<td>2 EM, 7 EM*, EM, 18 AI</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.001</td>
<td>2 EM, 6 EM*, 2 EM, 18 AI</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.005-0.01</td>
<td>3 EM, 5 EM*, 2 EM, 18 AI</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.05-0.1</td>
<td>4 EM, 4 EM*, 2 EM, 18 AI</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.6</td>
<td>7 EM, EM*, 3 EM, 18 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7</td>
<td>11 EM, 18 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8-0.9</td>
<td>12 EM, 17 AI</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>0.0001-0.001</td>
<td>2 EM, 2 EM*, EM, 25 AI</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.005-0.05</td>
<td>3 EM, EM*, EM, 25 AI</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.1</td>
<td>5 EM, 25 AI</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.8</td>
<td>6 EM, 19 AI</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9</td>
<td>7 EM, 20 AI</td>
<td>27</td>
</tr>
</tbody>
</table>
### CHAPTER 9. Unstructured G Model

<table>
<thead>
<tr>
<th>Setting</th>
<th>Data Set</th>
<th>Cut-off for ( p_{S(m)} )</th>
<th>Iteration Sequence</th>
<th>Number of Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>60%</td>
<td>1</td>
<td>0.0001</td>
<td>2 EM, 6 EM*, EM, 12 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.0005-0.01</td>
<td>2 EM, 6 EM*, 2 EM, 11 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.05-0.1</td>
<td>4 EM, 4 EM*, 2 EM, 11 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.6</td>
<td>6 EM, 2 EM*, 3 EM, 11 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7</td>
<td>7 EM, EM*, 4 EM, 10 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8-0.9</td>
<td>12 EM, 10 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.0001</td>
<td>2 EM, 3 EM*, EM, 12 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.0005-0.001</td>
<td>EM, 4 EM*, 2 EM, 11 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.005-0.01</td>
<td>2 EM, 3 EM*, 2 EM, 1 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.05-0.1</td>
<td>3 EM, 2 EM*, 2 EM, 11 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.7</td>
<td>8 EM, 11 AI</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8-0.9</td>
<td>9 EM, 10 AI</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.0001</td>
<td>EM, 7 EM*, 2 EM, 11 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.0005-0.01</td>
<td>2 EM, 6 EM*, 2 EM, 11 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.05</td>
<td>3 EM, 5 EM*, 3 EM, 10 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.1</td>
<td>4 EM, 4 EM*, 3 EM, 10 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.6</td>
<td>6 EM, 3 EM*, 3 EM, 10 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7</td>
<td>7 EM, 2 EM*, 3 EM, 10 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8</td>
<td>12 EM, 10 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9</td>
<td>13 EM, 10 AI</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.0001</td>
<td>EM, 5 EM*, EM, 13 AI</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.0005-0.0005</td>
<td>EM, 5 EM*, 2 EM, 13 AI</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.001</td>
<td>EM, 5 EM*, EM, 13 AI</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.005-0.01</td>
<td>2 EM, 4 EM*, 2 EM, 13 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.05-0.1</td>
<td>3 EM, 3 EM*, 2 EM, 13 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.6</td>
<td>5 EM, EM*, 3 EM, 12 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7-0.8</td>
<td>9 EM, 12 AI</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9</td>
<td>10 EM, 12 AI</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0.0001</td>
<td>EM, 4 EM*, 13 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.0005-0.001</td>
<td>EM, 4 EM*, EM, 12 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.005-0.05</td>
<td>2 EM, 3 EM*, EM, 12 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.1</td>
<td>3 EM, 2 EM*, EM, 12 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>4 EM, EM*, 2 EM, 11 AI</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6-0.9</td>
<td>7 EM, 11 AI</td>
<td>18</td>
</tr>
</tbody>
</table>
## 9.4. Multi-Environment Plant Variety Data

<table>
<thead>
<tr>
<th>Setting</th>
<th>Data Set</th>
<th>Cut-off for $p_{S(m)}$</th>
<th>Iteration Sequence</th>
<th>Number of Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>60%</td>
<td>6</td>
<td>0.0001-0.0005</td>
<td>EM, 8 EM*, EM, 14 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.001-0.01</td>
<td>2 EM, 7 EM*, EM, 14 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.05</td>
<td>3 EM, 6 EM*, 2 EM, 13 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.1</td>
<td>4 EM, 5 EM*, 2 EM, 13 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.6</td>
<td>6 EM, 3 EM*, 3 EM, 13 AI</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7</td>
<td>7 EM, 2 EM*, 3 EM, 13 AI</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8</td>
<td>8 EM, EM*, 4 EM, 12 AI</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9</td>
<td>13 EM, 12 AI</td>
<td>25</td>
</tr>
<tr>
<td>7</td>
<td>0.0001-0.005</td>
<td>2 EM, 8 EM*, 2 EM, 12 AI</td>
<td>24</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.01</td>
<td>3 EM, 7 EM*, 2 EM, 12 AI</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.05</td>
<td>3 EM, 7 EM*, 3 EM, 12 AI</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.1</td>
<td>4 EM, 6 EM*, 3 EM, 12 AI</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5-0.6</td>
<td>7 EM, 3 EM*, 4 EM, 11 AI</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7</td>
<td>8 EM, 2 EM*, 4 EM, 11 AI</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8</td>
<td>9 EM, EM*, 4 EM, 11 AI</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9</td>
<td>15 EM, 11 AI</td>
<td>26</td>
</tr>
<tr>
<td>8</td>
<td>0.0001</td>
<td>EM, 9 EM*, 2 EM, 10 AI</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.005-0.005</td>
<td>2 EM, 8 EM*, 2 EM, 10 AI</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.01</td>
<td>3 EM, 7 EM*, 2 EM, 10 AI</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.05</td>
<td>4 EM, 6 EM*, 2 EM, 10 AI</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.1</td>
<td>4 EM, 6 EM*, 3 EM, 10 AI</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.5</td>
<td>7 EM, 3 EM*, 3 EM, 11 AI</td>
<td>24</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.6</td>
<td>7 EM, 3 EM*, 4 EM, 10 AI</td>
<td>24</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.7</td>
<td>8 EM, 2 EM*, 4 EM, 10 AI</td>
<td>24</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.8</td>
<td>9 EM, EM*, 4 EM, 10 AI</td>
<td>24</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.9</td>
<td>15 EM, 9 AI</td>
<td>24</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0.0001</td>
<td>EM, 7 EM*, EM, 12 AI</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.0005-0.005</td>
<td>2 EM, 6 EM*, 2 EM, 11 AI</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.01</td>
<td>3 EM, 5 EM*, 2 EM, 11 AI</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.05-0.1</td>
<td>4 EM, 4 EM*, 2 EM, 11 AI</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.5</td>
<td>6 EM, 2 EM*, 3 EM, 13 AI</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.6</td>
<td>7 EM, EM*, 3 EM, 10 AI</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.7-0.9</td>
<td>12 EM, 10 AI</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.0001</td>
<td>AI, EM*, 3 EM, 7 EM*, 11 AI</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.0005-0.01</td>
<td>EM, 3 EM*, 11 AI</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.05-0.1</td>
<td>2 EM, 2 EM*, 11 AI</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.5</td>
<td>4 EM, EM*, 11 AI</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.6-0.9</td>
<td>5 EM, 11 AI</td>
<td>16</td>
<td></td>
</tr>
</tbody>
</table>
Table 9.34: The maximum cut-off for $p_{S(m)}$ that invoked EM iterations before an AI iteration failed for each of the data sets for each of the settings for the analysis of the simulated MET data sets using the EM/AI scheme with the score criterion.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>40%</th>
<th>50%</th>
<th>60%</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.9</td>
<td>0.9</td>
<td>0.8</td>
</tr>
<tr>
<td>2</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
</tr>
<tr>
<td>3</td>
<td>0.1</td>
<td>0.5</td>
<td>0.8</td>
</tr>
<tr>
<td>4</td>
<td>0.05</td>
<td>0.9</td>
<td>0.7</td>
</tr>
<tr>
<td>5</td>
<td>0.01</td>
<td>0.5</td>
<td>0.6</td>
</tr>
<tr>
<td>6</td>
<td>0.5</td>
<td>0.5</td>
<td>0.9</td>
</tr>
<tr>
<td>7</td>
<td>0.0001</td>
<td>0.005</td>
<td>0.9</td>
</tr>
<tr>
<td>8</td>
<td>0.05</td>
<td>&gt;0.9</td>
<td>0.9</td>
</tr>
<tr>
<td>9</td>
<td>0.005</td>
<td>0.7</td>
<td>0.7</td>
</tr>
<tr>
<td>10</td>
<td>0.05</td>
<td>0.1</td>
<td>0.6</td>
</tr>
</tbody>
</table>

We will now look at the results of the analyses of the simulated MET data sets with the uninformed starting values using all of the hybrid schemes, that is the EM/AI, PXEM/AI, local EM/AI and local PXEM/AI schemes with the AI, update and score criteria. We will use 3 internal local EM iterations for the local EM/AI scheme and 2 internal local PXEM iterations for the local PXEM/AI scheme. For the schemes with the update criterion we will use an EM type iteration if $\Delta^{(m)} > 0.1$. And for the schemes with the score criterion, we will use an EM type iteration if $p_{S(m)} < 0.95$. The results are presented in Table 9.35.

The results for the AI, update and score criteria were similar. Recall that the hybrid scheme with the AI criterion use AI iterations unless an AI iteration fails. If an AI iteration fails, the iteration sequence is restarted with an EM iteration. If a subsequent AI iteration fails, then an EM iteration is performed using the estimates from the previous EM type iteration. So the results from the analysis using the AI criterion indicate the minimum number of EM type iterations needed initially to obtain variance parameter estimates close enough to the solution to ensure convergence of the AI algorithm. Note that the hybrid schemes with the update and score criteria generally use a larger number of EM type iterations than the hybrid schemes with the AI criterion. However the extra EM type iterations take the variance parameter estimates closer to the solution and as a result fewer AI iterations are then required, resulting in the total number of iterations being similar for the three different criteria.

The PXEM/AI, local EM/AI and local PXEM/AI schemes outperformed the EM/AI scheme. For the majority of the data sets, the local PXEM/AI scheme converged in the fewest iterations, followed by the PXEM/AI scheme, then the local EM/AI scheme and finally the EM/AI scheme. Note that using the uninformed starting values, the number of EM iterations needed to obtain variance parameter estimates close enough to the solution for the AI algorithm to converge ranged from 2 to 10, whereas the number of PXEM iterations required ranged from 1 to 3, demonstrating the superior performance of the PXEM algorithm. The number of external local iterations needed to ensure convergence
Table 9.35: The number of iterations to convergence for the EM/AI, PXEM/AI, local EM/AI and local PXEM/AI schemes with the AI, update and score criteria for the simulated MET data sets using the uninformed starting values. 3 internal local EM iterations were used for the local EM/AI scheme and 2 internal local PXEM iterations were used for the local PXEM/AI scheme.

For the hybrid schemes with the update criterion, an EM type iteration was used if $\Delta(m) > 0.1$. For the hybrid schemes with the score criterion, an EM type iteration was used if $p_S(m) < 0.95$.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Data set</th>
<th>Hybrid scheme</th>
<th>AI criterion</th>
<th>Update criterion</th>
<th>Score criterion</th>
</tr>
</thead>
<tbody>
<tr>
<td>25%</td>
<td>1</td>
<td>EM/AI</td>
<td>27 (7/20)</td>
<td>27 (13/14)</td>
<td>26 (10/16)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>20 (2/17)</td>
<td>18 (4/14)</td>
<td>17 (3/14)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>19 (3/16)</td>
<td>20 (5/15)</td>
<td>19 (4/15)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>17 (1/16)</td>
<td>17 (2/15)</td>
<td>17 (2/15)</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>EM/AI</td>
<td>27 (5/22)</td>
<td>28 (10/18)</td>
<td>26 (7/19)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>21 (2/19)</td>
<td>17 (3/14)</td>
<td>21 (2/19)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>23 (2/21)</td>
<td>21 (4/17)</td>
<td>21 (3/19)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>20 (1/19)</td>
<td>19 (2/17)</td>
<td>20 (1/19)</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>EM/AI</td>
<td>42 (4/38)</td>
<td>48 (12/36)</td>
<td>45 (9/36)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>41 (1/40)</td>
<td>37 (3/34)</td>
<td>32 (2/30)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>25 (2/23)</td>
<td>40 (4/36)</td>
<td>39 (3/36)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>29 (1/28)</td>
<td>36 (2/34)</td>
<td>29 (1/28)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>EM/AI</td>
<td>23 (5/17)</td>
<td>24 (13/11)</td>
<td>23 (12/11)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>14 (2/12)</td>
<td>14 (4/10)</td>
<td>13 (3/10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>16 (2/14)</td>
<td>16 (5/11)</td>
<td>15 (5/11)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>13 (1/12)</td>
<td>12 (2/10)</td>
<td>12 (2/10)</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>EM/AI</td>
<td>16 (3/13)</td>
<td>18 (8/10)</td>
<td>17 (6/11)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>13 (1/12)</td>
<td>12 (3/9)</td>
<td>12 (2/10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>15 (1/14)</td>
<td>13 (3/10)</td>
<td>13 (3/10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>11 (1/10)</td>
<td>12 (2/10)</td>
<td>11 (1/10)</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>EM/AI</td>
<td>23 (5/18)</td>
<td>24 (12/12)</td>
<td>23 (11/12)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>19 (2/17)</td>
<td>17 (5/12)</td>
<td>16 (6/12)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>15 (1/4)</td>
<td>13 (2/11)</td>
<td>13 (2/11)</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>EM/AI</td>
<td>20 (2/18)</td>
<td>22 (8/14)</td>
<td>20 (6/14)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>16 (1/15)</td>
<td>18 (4/14)</td>
<td>15 (2/13)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>16 (1/15)</td>
<td>17 (3/14)</td>
<td>16 (2/14)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>14 (1/13)</td>
<td>16 (2/14)</td>
<td>14 (1/13)</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>EM/AI</td>
<td>34 (4/30)</td>
<td>30 (8/22)</td>
<td>30 (8/22)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>26 (2/24)</td>
<td>22 (3/19)</td>
<td>16 (2/24)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>28 (2/26)</td>
<td>24 (3/21)</td>
<td>24 (3/21)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>25 (1/24)</td>
<td>24 (2/22)</td>
<td>24 (2/22)</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>EM/AI</td>
<td>19 (3/16)</td>
<td>21 (9/12)</td>
<td>21 (8/13)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>15 (1/14)</td>
<td>15 (3/12)</td>
<td>14 (2/12)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>17 (1/16)</td>
<td>15 (3/12)</td>
<td>15 (3/12)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>13 (1/12)</td>
<td>13 (2/11)</td>
<td>13 (1/12)</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>EM/AI</td>
<td>18 (3/15)</td>
<td>27 (16/11)</td>
<td>19 (7/12)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>14 (1/13)</td>
<td>16 (6/10)</td>
<td>24 (1/12)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>16 (1/15)</td>
<td>16 (6/10)</td>
<td>15 (3/12)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>13 (1/12)</td>
<td>13 (3/10)</td>
<td>13 (1/12)</td>
</tr>
<tr>
<td>Setting</td>
<td>Data set</td>
<td>Hybrid scheme</td>
<td>AI criterion</td>
<td>Update criterion</td>
<td>Score criterion</td>
</tr>
<tr>
<td>---------</td>
<td>---------</td>
<td>---------------</td>
<td>--------------</td>
<td>-----------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>50%</td>
<td>1</td>
<td>EM/AI</td>
<td>24 (9/15)</td>
<td>31 (19/12)</td>
<td>25 (12/13)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>18 (3/15)</td>
<td>17 (6/11)</td>
<td>17 (4/13)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>19 (3/16)</td>
<td>19 (7/12)</td>
<td>17 (5/12)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>15 (2/13)</td>
<td>14 (3/11)</td>
<td>15 (2/13)</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>EM/AI</td>
<td>17 (4/23)</td>
<td>29 (10/19)</td>
<td>26 (6/20)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>24 (1/23)</td>
<td>23 (4/19)</td>
<td>21 (2/19)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>23 (2/21)</td>
<td>23 (4/19)</td>
<td>23 (3/20)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>21 (1/19)</td>
<td>21 (2/19)</td>
<td>20 (1/19)</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>EM/AI</td>
<td>24 (7/17)</td>
<td>27 (13/14)</td>
<td>26 (11/15)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>16 (2/14)</td>
<td>17 (4/13)</td>
<td>16 (3/13)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>23 (2/21)</td>
<td>18 (5/13)</td>
<td>18 (4/14)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>14 (1/13)</td>
<td>15 (2/13)</td>
<td>15 (2/13)</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>EM/AI</td>
<td>28 (10/18)</td>
<td>34 (19/15)</td>
<td>29 (14/15)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>18 (3/15)</td>
<td>18 (5/13)</td>
<td>18 (4/14)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>22 (4/18)</td>
<td>22 (7/15)</td>
<td>21 (6/15)</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>EM/AI</td>
<td>24 (7/17)</td>
<td>27 (13/14)</td>
<td>26 (11/15)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>16 (2/14)</td>
<td>17 (4/13)</td>
<td>16 (3/13)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>23 (2/21)</td>
<td>18 (5/13)</td>
<td>18 (4/14)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>14 (1/13)</td>
<td>15 (2/13)</td>
<td>15 (2/13)</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>EM/AI</td>
<td>29 (7/22)</td>
<td>30 (15/15)</td>
<td>30 (11/19)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>23 (2/21)</td>
<td>23 (5/18)</td>
<td>22 (3/19)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>25 (3/22)</td>
<td>21 (6/15)</td>
<td>21 (6/15)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>23 (1/22)</td>
<td>21 (3/18)</td>
<td>18 (2/16)</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>EM/AI</td>
<td>22 (2/20)</td>
<td>27 (10/17)</td>
<td>22 (6/16)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>19 (1/18)</td>
<td>22 (4/18)</td>
<td>20 (2/18)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>21 (1/20)</td>
<td>21 (4/17)</td>
<td>20 (3/17)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>19 (1/18)</td>
<td>20 (2/18)</td>
<td>19 (1/18)</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>EM/AI</td>
<td>26 (8/18)</td>
<td>31 (18/13)</td>
<td>27 (11/16)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>19 (2/17)</td>
<td>20 (6/14)</td>
<td>18 (3/15)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>21 (3/18)</td>
<td>20 (7/13)</td>
<td>19 (5/14)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>18 (1/17)</td>
<td>18 (3/15)</td>
<td>16 (2/14)</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>EM/AI</td>
<td>31 (8/23)</td>
<td>37 (22/15)</td>
<td>29 (12/17)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>22 (2/20)</td>
<td>18 (6/12)</td>
<td>20 (3/17)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>22 (3/19)</td>
<td>22 (8/14)</td>
<td>21 (4/17)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>20 (1/19)</td>
<td>16 (3/13)</td>
<td>17 (2/15)</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>EM/AI</td>
<td>34 (4/30)</td>
<td>32 (10/22)</td>
<td>29 (8/21)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>32 (1/31)</td>
<td>31 (4/27)</td>
<td>26 (2/24)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>23 (2/21)</td>
<td>25 (4/21)</td>
<td>22 (3/19)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>22 (1/21)</td>
<td>29 (2/27)</td>
<td>22 (1/21)</td>
</tr>
<tr>
<td>Setting</td>
<td>Data set</td>
<td>Hybrid scheme</td>
<td>AI criterion</td>
<td>Update criterion</td>
<td>Score criterion</td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
<td>---------------</td>
<td>--------------</td>
<td>------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>60%</td>
<td>1</td>
<td>EM/AI</td>
<td>22 (8/14)</td>
<td>25 (16/9)</td>
<td>21 (12/9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>12 (2/10)</td>
<td>12 (4/8)</td>
<td>12 (2/10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>11 (1/10)</td>
<td>10 (2/8)</td>
<td>11 (1/10)</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>EM/AI</td>
<td>18 (5/13)</td>
<td>27 (18/9)</td>
<td>19 (9/10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>14 (1/13)</td>
<td>13 (5/8)</td>
<td>11 (3/9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>14 (2/12)</td>
<td>15 (6/9)</td>
<td>14 (4/10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>11 (1/10)</td>
<td>11 (3/8)</td>
<td>10 (2/8)</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>EM/AI</td>
<td>23 (8/15)</td>
<td>26 (17/9)</td>
<td>23 (12/11)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>12 (2/10)</td>
<td>13 (5/8)</td>
<td>12 (3/9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>15 (3/12)</td>
<td>15 (6/9)</td>
<td>14 (5/9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>11 (1/10)</td>
<td>11 (3/8)</td>
<td>10 (2/8)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>EM/AI</td>
<td>22 (6/16)</td>
<td>28 (17/11)</td>
<td>19 (10/9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>13 (2/11)</td>
<td>13 (4/9)</td>
<td>13 (2/11)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>15 (3/12)</td>
<td>16 (6/10)</td>
<td>15 (4/11)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>12 (1/11)</td>
<td>11 (2/9)</td>
<td>12 (1/11)</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>EM/AI</td>
<td>18 (5/13)</td>
<td>25 (15/10)</td>
<td>19 (8/11)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>14 (1/13)</td>
<td>13 (5/8)</td>
<td>12 (2/10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>14 (2/12)</td>
<td>14 (5/9)</td>
<td>14 (3/11)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>11 (1/10)</td>
<td>11 (2/9)</td>
<td>11 (1/10)</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>EM/AI</td>
<td>24 (9/15)</td>
<td>31 (20/11)</td>
<td>26 (14/12)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>15 (2/13)</td>
<td>18 (6/12)</td>
<td>15 (2/13)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>18 (3/15)</td>
<td>18 (7/11)</td>
<td>17 (5/12)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>14 (1/13)</td>
<td>15 (3/12)</td>
<td>14 (1/13)</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>EM/AI</td>
<td>27 (10/17)</td>
<td>27 (16/11)</td>
<td>27 (17/10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>18 (2/16)</td>
<td>14 (5/9)</td>
<td>14 (3/11)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>20 (4/16)</td>
<td>18 (7/11)</td>
<td>18 (7/11)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>12 (2/10)</td>
<td>11 (3/9)</td>
<td>12 (2/10)</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>EM/AI</td>
<td>23 (10/13)</td>
<td>27 (18/9)</td>
<td>24 (14/10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>12 (3/9)</td>
<td>13 (5/8)</td>
<td>12 (3/9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>14 (4/10)</td>
<td>15 (6/9)</td>
<td>14 (5/9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>10 (2/8)</td>
<td>10 (2/8)</td>
<td>10 (2/8)</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>EM/AI</td>
<td>22 (8/13)</td>
<td>31 (21/10)</td>
<td>21 (12/9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>12 (2/10)</td>
<td>13 (4/9)</td>
<td>12 (3/9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>15 (3/12)</td>
<td>17 (7/10)</td>
<td>14 (5/9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>11 (1/10)</td>
<td>11 (2/9)</td>
<td>11 (1/10)</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>EM/AI</td>
<td>16 (3/13)</td>
<td>23 (13/10)</td>
<td>17 (6/11)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>12 (1/11)</td>
<td>13 (4/9)</td>
<td>12 (2/10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>14 (1/13)</td>
<td>14 (5/9)</td>
<td>13 (2/11)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>10 (1/9)</td>
<td>11 (2/9)</td>
<td>10 (1/9)</td>
</tr>
</tbody>
</table>
of the AI algorithm ranged from 1 to 4 for the local EM/AI scheme, and from 1 to 2 for the local PXEM/AI scheme. The local EM/AI scheme outperformed the EM/AI scheme, and the local PXEM/AI scheme outperforms the PXEM/AI scheme. The strong performance of the local schemes indicates that the convergence difficulties were in relation to the parameters in $G$ rather than $\sigma^2$.

We will now look at the results of the hybrid schemes for the simulated MET data sets using the informed starting values. The results are presented in Table 9.36. As expected, most of the schemes converged in fewer iterations using the informed starting values compared to the results using the uninformed starting values. For most of the data sets, when the local schemes worked, they converged in the fewest iterations, followed by the PXEM/AI scheme and then the EM/AI scheme. There was less of a difference in the number of iterations to convergence between all of the schemes using the informed starting values compared to the uninformed starting values. The results indicate that even though the hybrid schemes unnecessarily invoke EM type iterations, the total number of iterations to convergence for the hybrid schemes is similar to that of the AI algorithm.

The hybrid local schemes with the score criterion failed for all data sets using the informed starting values, whereas they successfully converged to the REML estimates using the uninformed starting values. The local schemes with the score criterion failed due to the local iterations only updating the elements of $G_p$ and holding $\sigma^2$ at the starting value. This resulted in the score function for $\sigma^2$ being large, resulting in a large value for the score statistic. Hence $p_{S(m)}$ was less than 0.95 for all iterations and the algorithm was unable to switch to AI iterations. To allow the algorithm to switch to AI iterations, the cut-off had to be lowered to less than $1^{-10}$. As we have seen, a cut-off this low would fail to detect convergence problems of AI iterations if we were using uninformed starting values. The hybrid local schemes with the update criterion also failed for two of the data sets using the uninformed starting values. Note that the same schemes successfully converged when the uninformed starting values are used. When the cut-off was increased slightly to 0.11, the local schemes converged successfully for these data sets. The convergence problems with the local schemes could possibly be resolved by adjusting $\sigma^2$ after updating the elements of $G_p$ at each iteration using the methodology outlined in Result A.19.
Table 9.36: The number of iterations to convergence for the EM/AI, PXEM/AI, local EM/AI and local PXEM/AI schemes with the AI, update and score criteria for the simulated MET data sets using the informed starting values. 3 internal local EM iterations were used for the local EM/AI scheme and 2 internal local PXEM iterations were used for the local PXEM/AI scheme. For the hybrid schemes with the update criterion, an EM type iteration was used if $\Delta^{(m)} > 0.1$. For the hybrid schemes with the score criterion, an EM type iteration was used if $p_{S_{(m)}} < 0.95$.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Data set</th>
<th>Hybrid scheme</th>
<th>AI criterion</th>
<th>Update criterion</th>
<th>Score criterion</th>
</tr>
</thead>
<tbody>
<tr>
<td>40%</td>
<td>1</td>
<td>EM/AI</td>
<td>12</td>
<td>20 (7/13)</td>
<td>20 (2/18)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>12</td>
<td>23 (3/20)</td>
<td>15 (2/13)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>12</td>
<td>16 (2/14)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>12</td>
<td>13 (2/11)</td>
<td>fail</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>EM/AI</td>
<td>19</td>
<td>19 (3/16)</td>
<td>19 (2/17)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>19</td>
<td>18 (2/16)</td>
<td>17 (1/16)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>19</td>
<td>19 (1/18)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>19</td>
<td>18 (1/17)</td>
<td>fail</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>EM/AI</td>
<td>34</td>
<td>38 (3/35)</td>
<td>35 (1/34)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>34</td>
<td>36 (2/34)</td>
<td>34 (1/33)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>34</td>
<td>36 (1/35)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>34</td>
<td>33 (1/32)</td>
<td>fail</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>EM/AI</td>
<td>12</td>
<td>18 (8/10)</td>
<td>14 (3/11)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>12</td>
<td>13 (3/10)</td>
<td>12 (2/10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>12</td>
<td>13 (3/10)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>12</td>
<td>12 (2/10)</td>
<td>fail</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>EM/AI</td>
<td>11</td>
<td>12 (2/10)</td>
<td>11 (1/10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>11</td>
<td>11 (1/10)</td>
<td>11 (1/10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>11</td>
<td>11 (1/10)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>11</td>
<td>11 (1/10)</td>
<td>fail</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>EM/AI</td>
<td>12</td>
<td>16 (4/12)</td>
<td>14 (2/12)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>12</td>
<td>13 (2/11)</td>
<td>13 (2/11)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>12</td>
<td>12 (2/10)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>12</td>
<td>11 (1/10)</td>
<td>fail</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>EM/AI</td>
<td>15</td>
<td>19 (4/15)</td>
<td>17 (2/15)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>15</td>
<td>17 (3/14)</td>
<td>16 (2/14)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>15</td>
<td>16 (1/15)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>15</td>
<td>17 (2/15)</td>
<td>fail</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>EM/AI</td>
<td>24</td>
<td>25 (3/22)</td>
<td>25 (2/23)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>24</td>
<td>23 (2/21)</td>
<td>23 (2/21)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>24</td>
<td>22 (1/21)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>24</td>
<td>23 (1/22)</td>
<td>fail</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>EM/AI</td>
<td>11</td>
<td>13 (2/11)</td>
<td>13 (2/11)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>11</td>
<td>13 (2/11)</td>
<td>13 (1/12)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>11</td>
<td>12 (1/11)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>11</td>
<td>12 (1/11)</td>
<td>fail</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>EM/AI</td>
<td>13</td>
<td>24 (14/10)</td>
<td>14 (2/12)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>13</td>
<td>15 (5/10)</td>
<td>12 (1/11)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>13</td>
<td>20 (10/10)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>13</td>
<td>16 (6/10)</td>
<td>fail</td>
</tr>
<tr>
<td>Setting</td>
<td>Data set</td>
<td>Hybrid scheme</td>
<td>AI criterion</td>
<td>Update criterion</td>
<td>Score criterion</td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
<td>---------------</td>
<td>--------------</td>
<td>------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>50%</td>
<td>1</td>
<td>EM/AI</td>
<td>12</td>
<td>25 (14/11)</td>
<td>16 (3/13)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>12</td>
<td>16 (5/11)</td>
<td>15 (2/13)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>12</td>
<td>15 (4/11)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>12</td>
<td>13 (3/10)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>EM/AI</td>
<td>21</td>
<td>23 (4/19)</td>
<td>22 (2/20)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>21</td>
<td>22 (3/19)</td>
<td>21 (2/19)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>21</td>
<td>22 (2/20)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>21</td>
<td>21 (2/19)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>EM/AI</td>
<td>15</td>
<td>18 (4/14)</td>
<td>16 (2/14)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>15</td>
<td>15 (3/12)</td>
<td>15 (2/13)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>15</td>
<td>15 (1/14)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>15</td>
<td>14 (2/12)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>EM/AI</td>
<td>16</td>
<td>23 (9/14)</td>
<td>17 (2/15)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>16</td>
<td>17 (3/14)</td>
<td>16 (2/14)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>16</td>
<td>18 (3/15)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>16</td>
<td>16 (2/14)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>EM/AI</td>
<td>15</td>
<td>18 (4/14)</td>
<td>16 (2/14)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>15</td>
<td>15 (3/12)</td>
<td>15 (2/13)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>15</td>
<td>15 (1/14)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>15</td>
<td>14 (2/12)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>EM/AI</td>
<td>20</td>
<td>24 (9/15)</td>
<td>23 (4/19)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>20</td>
<td>22 (4/18)</td>
<td>19 (2/17)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>20</td>
<td>20 (2/18)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>20</td>
<td>21 (2/19)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>EM/AI</td>
<td>20</td>
<td>22 (4/18)</td>
<td>21 (2/19)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>20</td>
<td>21 (3/18)</td>
<td>20 (2/18)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>20</td>
<td>20 (1/19)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>20</td>
<td>20 (2/18)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>EM/AI</td>
<td>17</td>
<td>27 (13/14)</td>
<td>17 (2/15)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>17</td>
<td>20 (5/15)</td>
<td>16 (2/14)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>17</td>
<td>20 (6/14)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>17</td>
<td>18 (4/14)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>EM/AI</td>
<td>17</td>
<td>32 (17/15)</td>
<td>21 (5/16)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>17</td>
<td>18 (5/13)</td>
<td>18 (2/16)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>17</td>
<td>20 (6/14)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>17</td>
<td>17 (3/14)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>EM/AI</td>
<td>30</td>
<td>33 (4/29)</td>
<td>31 (2/29)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>30</td>
<td>25 (3/22)</td>
<td>28 (2/26)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>30</td>
<td>30 (1/29)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>30</td>
<td>26 (2/24)</td>
<td>fail</td>
</tr>
<tr>
<td>Setting</td>
<td>Data set</td>
<td>Hybrid scheme</td>
<td>AI criterion</td>
<td>Update criterion</td>
<td>Score criterion</td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
<td>---------------</td>
<td>--------------</td>
<td>------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>60%</td>
<td>1</td>
<td>EM/AI</td>
<td>10</td>
<td>12 (4/8)</td>
<td>10 (2/8)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>10</td>
<td>11 (3/8)</td>
<td>10 (2/8)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>10</td>
<td>10 (1/9)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>10</td>
<td>10 (2/8)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>EM/AI</td>
<td>10</td>
<td>14 (5/9)</td>
<td>11 (2/9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>10</td>
<td>11 (3/8)</td>
<td>11 (2/9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>10</td>
<td>10 (1/9)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>10</td>
<td>v (2/7)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>EM/AI</td>
<td>10</td>
<td>12 (4/8)</td>
<td>10 (2/8)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>10</td>
<td>11 (3/8)</td>
<td>10 (2/8)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>10</td>
<td>9 (1/8)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>10</td>
<td>9 (2/7)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>EM/AI</td>
<td>11</td>
<td>14 (5/9)</td>
<td>12 (2/10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>11</td>
<td>12 (3/9)</td>
<td>11 (2/9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>11</td>
<td>fail</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>11</td>
<td>fail</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>EM/AI</td>
<td>10</td>
<td>14 (5/9)</td>
<td>11 (2/9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>10</td>
<td>(3/8) 11</td>
<td>(2/9) 11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>10</td>
<td>11 (2/9)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>10</td>
<td>10 (2/8)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>EM/AI</td>
<td>14</td>
<td>18 (6/12)</td>
<td>15 (2/13)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>14</td>
<td>16 (4/12)</td>
<td>14 (2/12)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>14</td>
<td>14 (2/12)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>14</td>
<td>14 (2/12)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>EM/AI</td>
<td>11</td>
<td>15 (5/10)</td>
<td>12 (2/10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>11</td>
<td>13 (4/9)</td>
<td>12 (2/10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>11</td>
<td>10 (1/9)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>11</td>
<td>10 (2/8)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>EM/AI</td>
<td>9</td>
<td>13 (5/8)</td>
<td>10 (2/8)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>9</td>
<td>11 (3/8)</td>
<td>10 (2/8)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>9</td>
<td>9 (1/8)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>9</td>
<td>9 (2/7)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>EM/AI</td>
<td>10</td>
<td>14 (5/9)</td>
<td>11 (2/9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>10</td>
<td>11 (3/8)</td>
<td>10 (2/8)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>10</td>
<td>fail</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>10</td>
<td>fail</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>EM/AI</td>
<td>11</td>
<td>13 (4/9)</td>
<td>12 (2/10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PXEM/AI</td>
<td>11</td>
<td>11 (2/9)</td>
<td>11 (1/10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local EM/AI</td>
<td>11</td>
<td>10 (1/9)</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>local PXEM/AI</td>
<td>11</td>
<td>10 (1/9)</td>
<td>fail</td>
</tr>
</tbody>
</table>
9.5 Robust Iterative Scheme

The results of the analyses of the random coefficient and MET data sets show that the AI algorithm was sensitive to the choice of starting values. If the starting values were too far from the REML solution, the AI algorithm failed to converge. In the cases where the AI algorithm converged, it did so very quickly. The EM and PXEM algorithms were stable, however they were very slow to converge. Although the PXEM algorithm was an improvement over the EM algorithm, it still fell well short of the AI algorithm when reasonable starting values were used. We were able to obtain reasonable starting values using Henderson’s Method III for the simulated MET data sets; however the starting values we used for the analysis of the random coefficient data sets were not optimal. We need a scheme that is robust to poor starting values, like the EM and PXEM algorithms, but that is also quick to converge like the AI algorithm.

The EM and PXEM updates were often closer to, or as close to, the REML solution as the AI updates over the early iterations, with the PXEM updates approaching the REML solution faster than the EM updates. Also, over the first few iteration, $r_e^{(m)}$ was often lower for the EM and PXEM algorithms than for the AI algorithm. However, once in the neighbourhood of the solution the EM and PXEM algorithms took increasingly smaller steps with each iteration, converging painfully slowly. So although the EM and PXEM algorithms were slower to converge than the AI algorithm, they were competitive in the early iterations. These results supported our use of a hybrid scheme that used EM type iterations initially and then switched to AI iterations once the variance parameter estimates were within the neighbourhood of the REML solution.

The EM/AI, PXEM/AI, local EM/AI and local PXEM/AI schemes performed well in the cases where the AI algorithm failed to converge. The local schemes failed to converge for some of the data sets due to the scheme not switching to AI iterations and hence holding $\sigma^2$ fixed and only updating $\gamma$. The PXEM/AI scheme generally converged in 1 or 2 fewer iterations than the EM/AI scheme, however, as we discussed in Section 8.5, the EM/AI scheme can be implemented more efficiently than the PXEM/AI algorithm.

The update criterion weights all of the variance parameters equally, while the score criterion incorporates a weighting scheme that takes into account the precision with which the variance parameter estimates are known. The convergence problems encountered when using the AI algorithm are often related to the parameters in the random effects variance matrix $G$, rather than the error variance matrix $R$ (Cullis et al., 2004). These parameters typically have larger variances, and so, as we did in Section 8.5, we will use the score criterion for our final robust scheme as it gives more weight to the variance parameters with larger variances. A cut-off of 0.5 for $p_{S(m)}$ was able to detect convergence problems before they occurred for the random coefficient data sets, however a cut-off of 0.95 was needed for the simulated MET data sets. When a cut-off of 0.5 was used for these data sets, some EM iterations were invoked due to the variance parameter updates being outside the parameter space (i.e. the variance parameter estimates resulted in a non-positive definite AI matrix), rather than due to $p_{S(m)} < 0.5$, however the scheme converged in the same number or fewer iterations than the scheme with a cut-off of 0.95. For our final robust scheme we will use a cut-off of 0.5 for $p_{S(m)}$. 
To determine an optimal iterative strategy we will look at the 4 different variations of the EM/AI scheme. We will compare the results obtained using two different definitions of the parameter space; all values of the variance parameters such that $\mathcal{I}_A(\kappa) > 0$ versus all values of the variance parameters such that $G$ is positive definite and $\sigma^2 > 0$. We will also compare the results using two different ways of calculating an EM iteration; calculating an EM update using the initial starting values or the estimates from the previous EM iteration (referred to as ”reset”) versus calculating an EM update using the current estimates (referred to as ”current”).

We will compare the results of these robust algorithms to the current methods used in popular software. ASReml and the REML directive in GenStat use the AI algorithm, and so we will compare the performance of our robust schemes to the AI algorithm. The `lmer` function in the `lme4` package in R uses 15 EM iterations and then switches to Newton Raphson iterations. We use the AI algorithm in preference to the Newton Raphson algorithm, and will compare the performance of our robust schemes to a scheme that uses 15 EM iterations and then switches to AI iterations (referred to as ”lmer”). Finally, we will also compare the performance of our robust schemes to the AI algorithm with informed starting values obtained using Henderson’s Method III. Table 9.37 presents the number of iterations to convergence for each of these various algorithms for the Orthodontic and Ultrafiltration data sets, the simulated random coefficient data sets and the simulated MET data sets.

For the random coefficient data sets, when the parameter space was defined as all values of the variance parameters such that $\mathcal{I}_A(\kappa) > 0$, the EM iterations invoked for the ”reset” and ”current” schemes were all invoked due to $p_S(m) < 0.5$. In almost all cases where the score criterion invoked an EM iteration, it was invoked in the first iteration, and subsequent EM iterations were invoked directly following a previous EM iteration. Hence the iteration sequence for the scheme that used the current updates to calculate EM iterations was equivalent to the iteration sequence of the scheme that reset to the initial starting values or the updates from the previous EM iteration. However, this was not the case for data sets 8 and 9 of the random coefficient data sets simulated using $\gamma = (1, 0.5, 1)'$ and data sets 4 and 5 of the random coefficient data sets simulated using $\gamma = (1, 0.8, 1)'$.

For data sets 9 of the random coefficient data sets simulated using $\gamma = (1, 0.5, 1)'$ and data sets 4 and 5 of the random coefficient data sets simulated using $\gamma = (1, 0.8, 1)'$, the iteration sequence began with one or two EM iterations, then switched to AI iterations, however on the second AI iteration $p_S(m)$ was less than 0.5 and so an EM iteration was done using the updates from the first AI iteration. After a few EM iterations, the scheme then successfully switched back to AI iterations. Recall that each iteration begins with an AI iteration. If the AI updates are outside of the parameter space (i.e. the variance parameter estimates result in a non-positive definite AI matrix) or if $p_S(m) < 0.5$, an EM iteration is done in place of the AI iteration. Calculating the EM iteration using the components already calculated for the failed AI iteration is computationally simpler than resetting to the previous EM iteration. For these data sets, although not resetting to the updates from the previous EM iteration resulted in a few extra iterations, it did not drastically increase the number of iterations to convergence. However, for data set 8 of the random coefficient data sets simulated using $\gamma = (1, 0.5, 1)'$, the iteration sequence
started with an AI iteration but then switched to EM iterations from this point on due to \( p_{G(m)} < 0.5 \), and failed to converge within 500 iterations. Whereas when the EM iteration was calculated using the initial starting values rather than the values from the AI iteration, only one EM iteration was required and the scheme converged in 7 iterations. Note that when the maximum number of allowable iterations was increased to 10000 for the scheme that used the current variance parameter updates to calculate EM iterations, the "current" scheme still failed to converged within 10000 iterations. Hence for this data set, it was necessary to reset to the initial starting values when an EM iteration was invoked.

For the simulated MET data sets, when the parameter space was defined as all values of the variance parameters such that \( I_A(\kappa) > 0 \), the results of the "reset" and "current" schemes were not equivalent for the majority of the data sets. For all data sets, the first EM iteration was invoked in the first iteration due to \( p_{G(1)} < 0.5 \). The majority of the subsequent EM iterations were invoked due to \( p_{G(m)} < 0.5 \), however a few were invoked due to \( I_A(\kappa^{(m)}) < 0 \). Not all EM iterations were invoked consecutively; in a number of cases, the scheme would switch back to EM iterations following an AI iteration. When this happened, the "reset" scheme reset to the variance parameter estimates from the previous EM iteration. However the "current" scheme used the variance parameter estimates from the previous AI iteration to calculate the EM iteration. For some of the data sets, not resetting to the variance parameter updates from the previous EM iteration did not result in a large number of extra iterations, and in some cases even resulted in fewer iterations. However for other data sets, using the current AI variance parameter estimates resulted in a significant increase in the number iterations to convergence, with the scheme failing to converge within 500 iterations for 4 of the data sets. Although a scheme that uses the current variance parameter estimates is computationally simpler, in order to ensure that our robust scheme will converge quickly in all cases, it is necessary to reset to the variance parameter estimates from the previous EM iteration.

For the "current" scheme when the parameter space was defined as all values of the variance parameters such that \( G > 0 \) and \( \sigma^2 > 0 \), the first EM iteration was always invoked in the first iteration and the subsequent EM iterations were invoked directly following a previous EM iteration, for all of the random coefficient and MET data sets. Hence the results for the "current" scheme were equivalent to the "reset" scheme. For the simulated MET data sets, all of the EM iterations were invoked due to \( G^{(m)} \) not being positive definite. For the random coefficient data sets, most of the EM iterations invoked by these schemes were invoked due to \( G^{(m)} \) not being positive definite, however some were invoked due to \( p_{G(m)} < 0.5 \).

For the simulated MET data sets for which the AI algorithm converged to a solution such that \( \hat{G} < 0 \), each AI iteration attempted by the "reset" and "current" schemes with the parameter space defined as all values of the variance parameters such that \( G > 0 \) and \( \sigma^2 > 0 \), resulted in updates such that \( G^{(m)} \) was not positive definite and hence an EM iteration was used. These schemes did not converge within 500 EM iterations. Recall that for each of these data sets, when the solution obtained using the AI algorithm was used as starting values for the EM algorithm, the EM algorithm converged to a solution such that \( \hat{G} \) was positive semi-definite and hence on the boundary of the parameter space.
For data set 1 of the simulated MET data sets with 40% of the observations missing, the "reset" and "current" schemes (with the parameter space defined as all values of the variance parameters such that $G > 0$ and $\sigma^2 > 0$) also required an EM update at each iteration, and failed to converge within 500 EM iterations. The solution obtained using the AI algorithm for this data set was such that $\hat{G}$ had a very small eigenvalue (0.00008) indicating that the solution was near the boundary of the parameter space. It is not surprising that the EM algorithm using the uninformed starting values did not converge within 500 iterations for these data sets as the EM algorithm is known to converge very slowly on or near the boundary of the parameter space (Laird and Ware, 1982).

Although defining the parameter space as all values of $G > 0$ and $\sigma^2 > 0$ is a more intuitive approach as it is based on the properties of the model rather than the algorithm, for a number of the simulated MET data sets it results in a scheme that requires a large number of EM iterations to obtain variance parameter estimates such that $G^{(m)} > 0$ to allow the scheme to switch to AI iterations to converge more rapidly. The scheme for which the parameter space was defined as all values of the variance parameters such that $I_A(\kappa) > 0$ and that reset to the initial starting values for the first EM iteration and to the previous EM estimates for subsequent EM iterations, converged in far fewer iterations for these data sets. Hence our preferred robust scheme uses AI iterations unless the AI iteration results in $I_A(\kappa) < 0$ or $\sigma^2^{(m)} < 0$, or if $p_{S(m)} < 0.5$, in which case an EM iteration is done in place of the AI iteration using the initial starting values for the first EM iteration or the estimates from the previous EM iteration for subsequent EM iterations.

Now we will look at how this robust scheme compared to popular current approaches. For the random coefficient data sets, using the uninformed starting values, our chosen robust scheme converged quickly in the cases where the AI algorithm failed to converge, and converged in a similar number of iterations to the AI algorithm for the cases where the AI algorithm converged. For the simulated MET data sets, the AI algorithm using the uninformed starting values failed to converge for all data sets. Our chosen robust scheme converged reasonably quickly for each data set, with the number of iterations to convergence ranging from 16 to 35. Therefore the results show that our robust algorithm is an improvement on the AI algorithm used in GenStat and ASReml.

For the random coefficient data sets, our robust scheme converged in less than half the number of iterations of the lmer scheme. For eight of the data sets, EM iterations were not invoked. For fourteen of the data sets, a single EM iteration was all that was needed to ensure that the AI algorithm would converge rapidly. Four of the data sets required 2 EM iterations, three required 3 EM iterations and one required 5 EM iterations. Hence the 15 EM iterations used by the lmer function in the lme4 package in R are excessive. For the simulated MET data sets, our robust scheme converged in between 1 and 16 iterations less than the lmer scheme, indicating that it is an improvement over the lmer scheme.

Finally we compare the results obtained using our robust scheme to those obtained using the AI algorithm with informed starting values. Recall that for the random coefficient data sets the AI algorithm using the informed starting values, obtained using the method outlines in Section 9.2.7.2, failed to converge for 4 of the data sets indicating that the
informed starting values used here were not optimal. Our robust scheme using the uninformed starting values converged rapidly for these data sets. For the data sets for which the AI algorithm with the informed starting values converged, our robust scheme converged in a similar number of iterations.

For the simulated MET data sets, the AI algorithm with the informed starting values obtained using Henderson’s Method III, converged in between 1 and 14 fewer iterations than our robust scheme in all but one case in which our robust scheme converged in 5 less iterations than the AI algorithm. This demonstrates that using Henderson’s Method III for models with $G = G_p \otimes I_m$ and $R = \sigma^2 I_n$ is a good way to obtain starting values. However, more complicated structures are often used for $R$ in the analysis of multi-environment plant variety trial and multi-trait animal breeding data. The extension of Henderson’s Method III for more general $R$ structures is an area for possible future research. Note that for models that have a large number of effects, Henderson’s Method III requires the computation of generalised inverses of matrices of very large dimensions (Searle, 1971) and may be computationally prohibitive. Although our robust scheme required a few more iterations, it still converged rapidly from the uninformed starting values. Hence in circumstances where it is difficult to obtain good starting values, our hybrid scheme combines the stable yet rapid progress of the EM algorithm in the early iterations with the speed of convergence of the AI algorithm in the neighbourhood of the REML solution to provide a hybrid scheme that is fast while being robust to choice of starting values.
Table 9.37: The number of iterations to convergence for the Orthodontic and Ultrafiltration data, the simulated random coefficient data sets and the simulated MET data sets. The results are for the AI algorithm, lmer scheme, 4 different versions of the robust hybrid EM/AI schemes with the score criterion with a cut-off of 0.5 for $p_{S(m)}$, all using the uninformed starting values of $G^{(0)} = I$ and $\sigma_{2}^{(0)}$, and the AI algorithm using informed starting values obtained using the method described in Section 9.2.7.2 for the random coefficient data sets or using Henderson’s Method III for the simulated MET data sets. ”RC 1”, ”RC 2” and ”RC 3” denote the simulated random coefficient data sets in simulated using $\gamma = (1, 0, 1)'$ and $\sigma^{2} = 1$, $\gamma = (1, 0.5, 1)'$ and $\sigma^{2} = 1$, and $\gamma = (1, 0.8, 1)'$ and $\sigma^{2} = 1$ respectively. ”MET 40%”, ”MET 50%” and ”MET 60%” denote the simulated MET data sets with 40%, 50% and 60% respectively. The results are presented in the form ”20 (15/5)” where 20 is the total number of iterations, 15 is the number of EM iterations and 5 is the number of AI iterations. A ”*” indicates that the scheme switched between EM and AI iterations a number of times. The rows in italics are the MET data sets for which the AI algorithm converged to a solution such that $\hat{G} < 0$, and the EM and PXEM algorithms (using the solution from the AI algorithm) converged to a solution such that $\hat{G} \geq 0$.

<table>
<thead>
<tr>
<th>data set</th>
<th>AI</th>
<th>lmer</th>
<th>$T_{M}(k^{(m)}) &gt; 0$</th>
<th>$G &gt; 0$ and $\sigma^{2} &gt; 0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ortho</td>
<td>fail</td>
<td>20 (15/5)</td>
<td>8 (3/5)</td>
<td>8 (3/5)</td>
</tr>
<tr>
<td>ultra</td>
<td>fail</td>
<td>22 (15/7)</td>
<td>13 (5/8)</td>
<td>14 (6/8)</td>
</tr>
<tr>
<td>RC 1</td>
<td>1</td>
<td>6</td>
<td>18 (15/3)</td>
<td>6 (0/6)</td>
</tr>
<tr>
<td>RC 2</td>
<td>1</td>
<td>6</td>
<td>19 (15/4)</td>
<td>6 (0/6)</td>
</tr>
<tr>
<td>RC 3</td>
<td>1</td>
<td>6</td>
<td>18 (15/4)</td>
<td>6 (0/6)</td>
</tr>
</tbody>
</table>
### Chapter 9. Unstructured G Model

**Table:**

<table>
<thead>
<tr>
<th>data set</th>
<th>AI</th>
<th>lmer</th>
<th>reset</th>
<th>current</th>
<th>reset</th>
<th>current</th>
<th>AI</th>
<th>HMI/III</th>
</tr>
</thead>
<tbody>
<tr>
<td>MET 40%</td>
<td>1</td>
<td>fail</td>
<td>28 (15/13)</td>
<td>22 (9/13)</td>
<td>29 (13/16)*</td>
<td>&gt;500 (&gt;500/0)</td>
<td>&gt;500 (&gt;500/0)</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>fail</td>
<td>28 (15/13)</td>
<td>26 (7/19)</td>
<td>198 (141/57)*</td>
<td>35 (19/16)</td>
<td>35 (19/16)</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>fail</td>
<td>51 (15/36)</td>
<td>35 (6/29)</td>
<td>40 (6/34)*</td>
<td>55 (19/36)</td>
<td>55 (19/36)</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>fail</td>
<td>26 (15/11)</td>
<td>20 (8/12)</td>
<td>20 (8/12)</td>
<td>&gt;500 (&gt;500/0)</td>
<td>&gt;500 (&gt;500/0)</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>fail</td>
<td>25 (15/10)</td>
<td>16 (4/12)</td>
<td>16 (4/12)</td>
<td>19 (9/10)</td>
<td>19 (9/10)</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>fail</td>
<td>27 (15/12)</td>
<td>21 (8/13)</td>
<td>19 (8/11)*</td>
<td>&gt;500 (&gt;500/0)</td>
<td>&gt;500 (&gt;500/0)</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>fail</td>
<td>29 (15/14)</td>
<td>18 (4/14)</td>
<td>18 (4/14)</td>
<td>18 (4/14)</td>
<td>18 (4/14)</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>fail</td>
<td>38 (15/23)</td>
<td>31 (6/25)</td>
<td>28 (6/22)*</td>
<td>31 (10/21)</td>
<td>31 (10/21)</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>fail</td>
<td>27 (15/12)</td>
<td>18 (5/13)</td>
<td>18 (5/13)</td>
<td>25 (13/12)</td>
<td>25 (13/12)</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>fail</td>
<td>26 (15/11)</td>
<td>18 (4/14)</td>
<td>18 (4/14)</td>
<td>22 (10/12)</td>
<td>22 (10/12)</td>
<td>13</td>
</tr>
<tr>
<td>MET 50%</td>
<td>1</td>
<td>fail</td>
<td>27 (15/12)</td>
<td>24 (11/13)</td>
<td>122 (94/28)*</td>
<td>46 (35/11)</td>
<td>46 (35/11)</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>fail</td>
<td>34 (15/19)</td>
<td>26 (5/21)</td>
<td>26 (6/20)*</td>
<td>28 (9/19)</td>
<td>28 (9/19)</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>fail</td>
<td>29 (15/14)</td>
<td>24 (9/15)</td>
<td>&gt;500 (&gt;423/77)*</td>
<td>212 (200/12)</td>
<td>212 (200/12)</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>fail</td>
<td>31 (15/16)</td>
<td>29 (13/16)</td>
<td>&gt;500 (&gt;439/61)*</td>
<td>286 (273/13)</td>
<td>286 (273/13)</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>fail</td>
<td>29 (15/14)</td>
<td>24 (9/15)</td>
<td>&gt;500 (&gt;423/77)*</td>
<td>212 (200/12)</td>
<td>212 (200/12)</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>fail</td>
<td>30 (15/15)</td>
<td>20 (9/20)</td>
<td>&gt;500 (&gt;64/75)*</td>
<td>123 (105/18)</td>
<td>123 (105/18)</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>fail</td>
<td>33 (15/18)</td>
<td>23 (4/19)</td>
<td>23 (4/19)</td>
<td>22 (6/16)</td>
<td>22 (6/16)</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>fail</td>
<td>29 (15/14)</td>
<td>26 (10/16)</td>
<td>20 (5/15)*</td>
<td>42 (28/14)</td>
<td>42 (28/14)</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>fail</td>
<td>31 (15/16)</td>
<td>29 (11/18)</td>
<td>67 (46/21)*</td>
<td>&gt;500 (&gt;500/0)</td>
<td>&gt;500 (&gt;500/0)</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>fail</td>
<td>38 (15/23)</td>
<td>25 (6/19)</td>
<td>31 (6/25)*</td>
<td>37 (14/23)</td>
<td>37 (14/23)</td>
<td>30</td>
</tr>
<tr>
<td>MET 60%</td>
<td>1</td>
<td>fail</td>
<td>25 (15/10)</td>
<td>22 (11/11)</td>
<td>23 (13/10)*</td>
<td>233 (225/8)</td>
<td>233 (225/8)</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>fail</td>
<td>24 (15/9)</td>
<td>19 (8/11)</td>
<td>27 (16/11)*</td>
<td>25 (16/9)</td>
<td>25 (16/9)</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>fail</td>
<td>24 (15/9)</td>
<td>22 (12/10)</td>
<td>23 (12/11)*</td>
<td>122 (114/8)</td>
<td>122 (114/8)</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>fail</td>
<td>26 (15/11)</td>
<td>21 (9/12)</td>
<td>38 (23/15)*</td>
<td>48 (39/9)</td>
<td>48 (39/9)</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>fail</td>
<td>25 (15/10)</td>
<td>18 (7/11)</td>
<td>28 (18/10)*</td>
<td>24 (14/10)</td>
<td>24 (14/10)</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>fail</td>
<td>27 (15/13)</td>
<td>23 (12/13)</td>
<td>25 (11/14)*</td>
<td>&gt;500 (&gt;500/0)</td>
<td>&gt;500 (&gt;500/0)</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>fail</td>
<td>26 (15/11)</td>
<td>23 (14/11)</td>
<td>23 (10/13)*</td>
<td>&gt;500 (&gt;500/0)</td>
<td>&gt;500 (&gt;500/0)</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>fail</td>
<td>24 (15/9)</td>
<td>23 (13/10)</td>
<td>21 (11/10)*</td>
<td>&gt;500 (&gt;500/0)</td>
<td>&gt;500 (&gt;500/0)</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>fail</td>
<td>24 (15/9)</td>
<td>21 (11/10)</td>
<td>25 (14/11)*</td>
<td>105 (97/8)</td>
<td>105 (97/8)</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>fail</td>
<td>24 (15/9)</td>
<td>16 (5/11)</td>
<td>16 (5/11)*</td>
<td>18 (7/11)</td>
<td>18 (7/11)</td>
<td>11</td>
</tr>
</tbody>
</table>
Maximisation of the REML log-likelihood to obtain estimates of the variance parameters in linear mixed models typically requires an iterative scheme. In this thesis, we have reviewed the AI, EM and PXEM algorithms. A good iterative method should converge to a global maximum for a wide range of starting values, at each iteration it should be relatively quick to calculate the update, and it should converge in relatively few iterations (Searle et al., 1992).

The AI algorithm is a Newton-Raphson type algorithm that replaces the observed information matrix by the average information matrix which is much easier to compute, as discussed in Chapter 3. This approximation results in the AI algorithm being a first-order scheme rather than a second order scheme like the Newton-Raphson algorithm. The closer the average information matrix is to the observed information matrix, the closer the convergence rate is to zero and hence approximate quadratic (superlinear) convergence is achieved. Like all NR type algorithms, the AI algorithm does not guarantee an increase in the log-likelihood with each iteration, nor does it guarantee to keep the variance parameters within the parameter space. The AI algorithm converges rapidly if the starting values are in the neighbourhood of the REML solution. However, when the starting values are too far from the REML solution, the AI algorithm can overshoot the solution or step in the wrong direction, producing variance parameter estimates that are outside of the parameter space and failing to converge to the REML solution.

The EM algorithm is guaranteed to keep the variance parameters within the parameter space, and each iteration is guaranteed to maintain or increase the log-likelihood, no matter what starting values are used. However as the number of variance parameters increases, it can be painfully slow to converge. The PXEM algorithm maintains the stability of the EM algorithm while yielding faster convergence. Our results demonstrate the improvement in convergence rate of the PXEM algorithm over the EM algorithm increases with model complexity. However our results also demonstrate that although the PXEM algorithm displays faster convergence than the EM algorithm, it still falls well short of the speed of the AI algorithm.

We evaluated the performance of these algorithms by considering the number of iterations to convergence and the rate of convergence. We have not considered the computational
burden of each iteration. Lindstrom and Bates (1988) showed that if the number of variance parameters was small, the computational burden of each iteration of a well implemented NR algorithm was not significantly greater than the cost of an EM iteration. Harville and Callanan (1991) and Callanan and Harville (1991) showed that for a simple variance components model with a single random effect the computing costs for the EM, NR and FS algorithms were small, with the NR and FS having slightly larger computing costs than EM. However they commented that in more complex models with a greater number of variance parameters, the NR and FS algorithms were likely to have significantly greater computing costs than the EM algorithm. The AI algorithm is a more computationally efficient algorithm than the NR and FS algorithms as it avoids the evaluation of the traces of large matrices which appear in both the observed and expected information matrices. Johnson and Thompson (1995) and Gilmour et al. (1995) demonstrated that the computational requirements for the AI algorithm using sparse matrix techniques (as implemented in ASReml) are similar to that of the EM algorithm for a range of models, including models for large data sets with complex variance structures.

A direct comparison of the computational burden between the AI and PXEM algorithms has not been published. However Foulley and van Dyk (2000) found that the computing time per PXEM iteration was double that of an EM iteration. Cullis et al. (2004) comment on the difficulty in implementing the PXEM algorithm in ASReml due to the calculation of the off-diagonal terms of the inverse of the coefficient matrix $C$ of the mixed model equations, namely $C^{-1}X^T$. Our results show that the number of iterations to convergence for the PXEM algorithm was similar to that of the EM algorithm for the variance components model and hence the total computational cost of the PXEM was greater than that of the EM algorithm. However the PXEM algorithm converged in considerably fewer iterations than the EM algorithm for some of the random coefficient data sets and most of the simulated MET data sets. Nevertheless, this improvement did not come close to matching that of the AI algorithm. Our results for the analysis of the simulated MET data sets show that the AI algorithm (with good starting values) converged in between 10 and 30 iterations, whereas the PXEM algorithm generally required at least 100 iterations and the EM algorithm over 500 iterations. Hence even if an AI iteration has a computing cost of twice that of an EM iteration, we are still much better off using the AI algorithm. Our results indicate that the AI algorithm is the most computationally efficient of the three algorithms.

In summary, the EM and PXEM algorithms are stable, that is they stay within the parameter space, however they can be slow to converge. The AI algorithm converges quickly provided that the starting values are within the neighbourhood of the REML solution, however if the starting values are too far from the solution, it fails to converge. The work in this thesis was motivated by problems in reliability of convergence encountered when fitting linear mixed models in ASReml using the AI algorithm. Our results have shown that the performance of the EM and PXEM was generally at least as good as that of the AI algorithm in the early iterations. The EM and PXEM variance parameter updates over the first few iterations were often closer to the REML solution than the AI updates. Our description of the AI algorithm as a first-order iterative scheme in Chapter 6 allows us to compute the theoretical and empirical linear rate of convergence for the AI algorithm and directly compare the convergence rate of the AI algorithm to that of the EM and PXEM algorithms. In doing so, we found that although the EM and PXEM
algorithms were much slower to converge, \( r_{(m)} \) was often lower over the first few iterations for the EM and PXEM algorithms than for the AI algorithm. However, once the variance parameter updates were in the neighbourhood of the REML solution, the EM and PXEM algorithms slowed down, taking increasingly smaller steps towards the solution, whereas the AI algorithm converged rapidly. These results support the use of a hybrid scheme that used EM-type iterations initially to obtain variance parameters estimates in the neighbourhood of the REML solution, and then switches to AI iterations, hence combining the stability of the EM-type algorithms with the rapid convergence of the AI algorithm.

We presented and compared EM/AI and PXEM/AI schemes, as well as local EM/AI and local PXEM/AI schemes in which the local iterations updated the random effects variance parameters only. The local schemes failed to converge for some of the data sets due to the scheme not switching to AI iterations and hence holding \( \sigma^2 \) fixed and only updating \( \gamma \). Further research needs to be done to see if this can be resolved by adjusting \( \sigma^2 \) after \( \gamma \) is updated at each iteration using Result A.19. The PXEM/AI scheme generally converged in slightly fewer iterations than the EM/AI scheme. However, computationally the EM/AI algorithm can be implemented more efficiently than the PXEM/AI algorithm. Hence our preferred hybrid iterative scheme is the EM/AI scheme.

The \texttt{lmer} function in \texttt{R} uses 15 EM iterations and then switches to NR iterations. \texttt{ASReml} has an option to use an EM iteration when an AI iteration results in variance parameter estimates outside of the parameter space. Rather than using a fixed number of EM-type iterations, or only using an EM-type iteration when an AI iteration fails, we investigated different techniques for determining when to use EM-type iterations and when to switch to AI iterations. The update criteria evaluated the relative difference between the current estimates and the previous estimates, the same measure used for the convergence criterion, and invoked an EM type iteration if this difference was too large. The score criterion calculated a proximity measure, \( S^{(m)} \) for the score vector to assess how close it was to the zero vector, invoking an EM iteration if \( p_{S^{(m)}} \) was too small. In our final robust iterative scheme we chose to use the score criterion as it takes into account the precision with which the variance parameter estimates are known. However our results indicate that both criteria worked well, invoking EM iterations when needed and then allowing the scheme to switch to AI iterations once the variance parameter estimates were in the neighbourhood of the REML solution.

To determine an optimal iterative strategy we looked at four different variations of the EM/AI scheme with the score criterion. We compared the results obtained using two different definitions of the parameter space: all values of the variance parameters such that the average information matrix is positive definite, versus all values of the variance parameters such that the variance matrices \( G \) and \( R \) are positive definite. For each of these schemes, we compared the results obtained by resetting to the initial starting values or the estimates from the previous EM iteration to calculate an EM update versus using the current estimates. For the analyses using the variance components model, the four schemes converged in the same number of iterations for all but one data set. And so we took the more intuitive approach of defining the parameter space as all values of the variance parameters such that the variance matrices \( G \) and \( R \) are positive definite for our recommended scheme. We also took the computationally more efficient approach of
using the current estimates to calculate EM iterations rather than resetting to the initial
starting values for the first EM iteration or to the estimates from the previous EM iteration
for subsequent EM iterations. For the analyses using the models with the unstructured
random effects variance matrix, taking the intuitive approach of defining the parameter
space as all values of the variance parameters such that the variance matrices $G$ and $R$ are
positive definite resulted in the scheme taking hundreds of iterations to converge for some
of the simulated MET data sets. To obtain a scheme that was robust to poor starting
values and able to converge quickly, it was necessary to define the parameter space as all
values of the variance parameters such that the average information matrix was positive
definite. It was also necessary to reset to the initial starting values when the first EM
iteration was invoked and to the estimates from the previous EM iteration for subsequent
EM iterations, rather than just using the current variance parameter estimates.

Hence to ensure successful and rapid converge regardless of the starting values for a wide
range of models, we recommend an EM/AI scheme that uses AI iterations unless the
AI iteration results in $\mathcal{I}_A(\kappa) < 0$ or $\sigma^2(m) < 0$, or if $p_{S(m)} < 0.5$, in which case an EM
iteration is done in place of the AI iteration using the initial starting values for the first EM
iteration or the estimates from the previous EM iteration for subsequent EM iterations.

Note that for the analysis of the simulated MET data sets, using our final robust scheme
with the score criterion with a cut-off of 0.5 for $p_{S(m)}$, some of the EM iterations were
invoked due to the variance parameter updates being outside the parameter space, rather
than due to $p_{S(m)} < 0.5$. This did not increase the number of iterations to convergence,
however the scheme could possibly be fine tuned by increasing the strictness of the criterion
as the iterations progress, to make sure that AI iterations are not invoked prematurely.
This may help to ensure that EM iterations are invoked sequentially and hence the current
variance parameter estimates can be used to calculate each EM iteration, rather than
resetting to the updates from the previous EM iteration. This would result in a more
computationally efficient scheme.

We compared the performance of our recommended scheme to some of the current ap-
proaches in popular software packages. By default, ASReml and the REML directive in
GenStat use the AI algorithm. Our hybrid scheme converged in relatively few iterations
for all cases where the AI algorithm failed to converge, demonstrating that our scheme
is robust to poor starting values unlike the AI algorithm. The lmer function in the lme4
package in R uses 15 EM iterations and then switches to NR iterations. In all cases our
scheme converged in fewer iterations than our version of the lmer scheme, that uses AI iter-
ations rather than NR iterations, with the difference increasing as the number of variance
parameters decreased. Hence we have demonstrated that the flexibility our scheme pro-
vides by assessing whether EM iterations are needed results in faster convergence where
needed.

Finally we compared the results obtained using our hybrid scheme to those obtained
using the AI algorithm with informed starting values. For the analyses using the variance
components model, the AI algorithm with starting values obtained using Henderson’s
Method III, outperformed our robust scheme in almost all cases, but only by 2 iterations
on average. The method used to obtain starting values for the analyses using the random
coefficient models was not optimal and hence the AI algorithm using these informed

Note that for the analysis of the simulated MET data sets, using our final robust scheme
with the score criterion with a cut-off of 0.5 for $p_{S(m)}$, some of the EM iterations were
invoked due to the variance parameter updates being outside the parameter space, rather
than due to $p_{S(m)} < 0.5$. This did not increase the number of iterations to convergence,
however the scheme could possibly be fine tuned by increasing the strictness of the criterion
as the iterations progress, to make sure that AI iterations are not invoked prematurely.
This may help to ensure that EM iterations are invoked sequentially and hence the current
variance parameter estimates can be used to calculate each EM iteration, rather than
resetting to the updates from the previous EM iteration. This would result in a more
computationally efficient scheme.

We compared the performance of our recommended scheme to some of the current ap-
proaches in popular software packages. By default, ASReml and the REML directive in
GenStat use the AI algorithm. Our hybrid scheme converged in relatively few iterations
for all cases where the AI algorithm failed to converge, demonstrating that our scheme
is robust to poor starting values unlike the AI algorithm. The lmer function in the lme4
package in R uses 15 EM iterations and then switches to NR iterations. In all cases our
scheme converged in fewer iterations than our version of the lmer scheme, that uses AI iter-
ations rather than NR iterations, with the difference increasing as the number of variance
parameters decreased. Hence we have demonstrated that the flexibility our scheme pro-
vides by assessing whether EM iterations are needed results in faster convergence where
needed.

Finally we compared the results obtained using our hybrid scheme to those obtained
using the AI algorithm with informed starting values. For the analyses using the variance
components model, the AI algorithm with starting values obtained using Henderson’s
Method III, outperformed our robust scheme in almost all cases, but only by 2 iterations
on average. The method used to obtain starting values for the analyses using the random
coefficient models was not optimal and hence the AI algorithm using these informed
starting values failed to converge for 4 of the data sets. Our scheme converged rapidly for all of the data sets using the uninformed starting values, converging in a similar number of iterations to the AI algorithm for the data sets for which the AI algorithm with the informed starting values converged. For the simulated MET data sets, the AI algorithm with the informed starting values obtained using Henderson’s Method III, converged in between 1 and 14 fewer iterations than our robust scheme in all but one case in which our robust scheme converged in 5 less iterations than the AI algorithm, thus demonstrating that Henderson’s Method III is a good method to obtain starting values for for models with\( G = G_p \otimes I_m \) and \( R = \sigma^2 I_n \). However for models that have a large number of effects, Henderson’s Method III requires the computation of generalised inverses of matrices of very large dimensions (Searle, 1971) and may be computationally prohibitive. Although our robust scheme required a few more iterations, it still converged rapidly using the uninformed starting values. We have demonstrated that the performance of our recommended is comparable to that of the AI algorithm with good starting values. Hence in circumstances where it is difficult to obtain good starting values, our hybrid scheme combines the stable yet rapid progress of the EM algorithm in the early iterations with the speed of convergence of the AI algorithm in the neighbourhood of the REML solution to provide a hybrid scheme that is fast while being robust to choice of starting values.

Several authors such as Jennrich and Sampson (1976), Harville (1977), Lindstrom and Bates (1988) and Meyer (2008) discuss alternative ways of improving the performance of Newton-Raphson type algorithms such as the AI algorithm. Dennis and Schnabel (1996) and Nocedal and Wright (1999) discuss in detail methods such as adjustment of the information matrix \( I(\kappa) \) to ensure that the algorithm is heading in the right direction, and step size adjustment to stop the algorithm from overshooting when it is heading in the right direction.

As discussed in Section 7.2, to ensure that a Newton type algorithm of the form
\[
\kappa^{(m+1)} = \kappa^{(m)} + I(\kappa^{(m)})^{-1} U(\kappa^{(m)})
\]
needs to be positive definite. If \( I(\kappa^{(m)}) \) is positive definite, \( I(\kappa^{(m)})^{-1} U(\kappa^{(m)}) \) is a descent direction and hence the algorithm will head toward the maximum. For the AI algorithm, we chose to terminate the iteration process if \( I_A(\kappa)^{(m)} \) became non-positive definite. In the context of the Newton-Raphson algorithm, Dennis and Schnabel (1996) and Nocedal and Wright (1999) discuss ways of modifying \( I(\kappa^{(m)}) \), such as the simple approach of adding the smallest possible scaled identity matrix to \( I(\kappa^{(m)}) \) such that \( I(\kappa^{(m)}) + \alpha I \) is positive definite, or a more sophisticated approach of modifying \( I(\kappa^{(m)}) \) by performing a Cholesky factorization of \( I(\kappa^{(m)}) \) and increasing the diagonal elements to ensure that they are sufficiently positive and hence the modified \( I(\kappa^{(m)}) \) is positive definite. It is important to make the smallest possible modification to \( I(\kappa^{(m)}) \) so that the speed of convergence of the algorithm is preserved.

Dennis and Schnabel (1996) and Nocedal and Wright (1999) also discuss methods to adjust the step size to stop the algorithm from overshooting, assuming that \( I(\kappa^{(m)}) \) is positive definite and hence the algorithm is heading in the right direction. A common procedure is to try a full step size, that is \( \kappa^{(m+1)} = \kappa^{(m)} + I(\kappa^{(m)})^{-1} U(\kappa^{(m)}) \). If the resulting update is not within the parameter space or does not result in an increase in the log-likelihood, then backtrack in a systematic way to determine the largest value of \( \lambda < 1 \) such that \( \kappa^{(m+1)} = \kappa^{(m)} + \lambda I(\kappa^{(m)})^{-1} U(\kappa^{(m)}) \) results in an update that it acceptable.
Again it is important to make the smallest modification possible so as not to decrease the rate of convergence once the algorithm is close to the solution. Note that ASReml modifies the step size to control potential overshooting with the AI algorithm, starting with a step size of 0.1, and increasing towards 1 with each iteration as the default.

We have demonstrated that our EM/AI scheme is able to converge rapidly using poor starting values without taking into account step length adjustment or modification of $\mathcal{I}_A(\kappa)$. When the AI matrix was not positive definite or the AI algorithm resulted in updates outside of the parameter space, we used EM iterations to obtain variance parameter estimates close enough to the REML solution to ensure rapid convergence of the AI algorithm.

The AI algorithm, as we have presented it, performs unconstrained optimisation of the REML log-likelihood. Our results for the analyses of the variance component model and random coefficient model data sets supported the use of a scheme that defined the parameter space as all values of the variance parameters such that the variance matrices are positive definite. However this definition of the parameter space was too strict for the analyses of the simulated MET data sets. As we saw in the analysis of the simulated MET data sets, the AI algorithm converged to a solution such that $\hat{G}$ was not positive definite for some of the data sets. Meyer and Smith (1996) and Meyer (2008) discuss the use of parameterisations of the variance matrices to ensure that the variance matrices remain positive definite during iterations. The most commonly used parameterisations are the Cholesky and the log-Cholesky parameterisations as outlined in Pinheiro and Bates (1996). Note that the Cholesky parameterisation is an option in ASReml and in the lmer package in R. Use of parameterisations to ensure that the variance matrices remain positive definite during iterations is a method that could be used to fine tune our proposed EM/AI scheme.
Appendix

Result A.1
Let $\mathbf{a}$ is a $n \times 1$ vector, $\mathbf{x}$ is a $n \times 1$ vector, $\mathbf{f}(\mathbf{x})$ is a $n \times 1$ vector, $\mathbf{A}$ is a $n \times n$ matrix and $\mathbf{B}$ is a $n \times n$ matrix. Define $A_i = \frac{\partial A}{\partial x_i}$. Then

\[
\frac{\partial a'x}{\partial x} = a
\]
\[
\frac{\partial a'f(x)}{\partial x} = \frac{\partial f(x)'}{\partial x}a
\]
\[
\frac{\partial x'A}{\partial x} = A
\]
\[
\frac{\partial x'Ax}{\partial x} = 2Ax
\]
\[
\frac{\partial f(x)'Af(x)}{\partial x} = 2\frac{\partial f(x)'}{\partial x}Af(x)
\]
\[
\frac{\partial (\mathbf{a} - \mathbf{A}\mathbf{x})'B(\mathbf{a} - \mathbf{A}\mathbf{x})}{\partial \mathbf{x}} = -2A'B(\mathbf{a} - \mathbf{A}\mathbf{x})
\]
\[
\frac{\partial \log |\mathbf{A}|}{\partial x_i} = \text{tr}(A^{-1} \dot{A}_i)
\]
\[
\frac{\partial A^{-1}}{\partial x_i} = -A^{-1} \dot{A}_i A^{-1}
\]
Result A.2
If $A$ is a $m \times m$ matrix, $B$ is a $m \times n$ matrix, $C$ is a $n \times m$ matrix and $D$ is a $n \times n$ matrix, then
\[
(A + BCD)^{-1} = A^{-1} - A^{-1}B(C^{-1} + DA^{-1}B)^{-1}DA^{-1}
\]

If $A$ is non-singular
\[
|A + BC| = |A||I_m + A^{-1}BC| = |A||I_n + CA^{-1}B|
\]

If $D$ is non-singular, and $M$ is given by
\[
M = \begin{bmatrix} A & B \\
C & D \end{bmatrix}
\]
then
\[
|M| = |D||A - BD^{-1}C|
\]

Proof
See Mardia et al. (1979)

Result A.3
\[
R^{-1}Z(Z'R^{-1}Z + G^{-1})^{-1} = (R + ZGZ')^{-1}ZG
\]

Proof
Using Result A.2
\[
(R + ZGZ')^{-1} = R^{-1} - R^{-1}Z(Z'R^{-1}Z + G^{-1})^{-1}Z'R^{-1}
\]
as long as $R$ and $G$ are non-singular. This can be rearranged to give
\[
(Z'R^{-1}Z + G^{-1})^{-1}Z'R^{-1} = GZ'(R + ZGZ')^{-1}
\]
and transposing gives
\[
R^{-1}Z(Z'R^{-1}Z + G^{-1})^{-1} = (R + ZGZ')^{-1}ZG
\]
Result A.4
If the matrices $X$ and $L_2$ are both of full column rank and satisfy $L'_2X = 0$ and $H$ is positive definite then
\[ H - HL_2(L'_2HL_2)^{-1}L'_2H = X(X'H^{-1}X)^{-1}X' \]
and rearranging gives
\[ L_2(L'_2HL_2)^{-1}L'_2 = H^{-1} - H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1} = P \]

Proof
See Verbyla (1990)

Result A.5

\[ P = R^{-1} - R^{-1}WC^{-1}W'R^{-1} \]

Proof
From Result A.2 we have
\[ (R + ZGZ')^{-1}ZG = R^{-1}Z(Z'R^{-1}Z + G^{-1})^{-1} \]

We can rewrite this as
\[ H^{-1}ZG = R^{-1}ZK \]
where $K = (Z'R^{-1}Z + G^{-1})^{-1}$. Result A.2 also allows us to write $H^{-1}$ in terms of $K$
\[ H^{-1} = (R + ZGZ')^{-1} = R^{-1} - R^{-1}Z(Z'R^{-1}Z + G^{-1})^{-1}Z'R^{-1} = R^{-1} - R^{-1}ZKZ'R^{-1} \]

We can also write the elements of $C^{-1}$ in (2.21) in terms of $K$
\[ C^{XZ} = -(X'H^{-1}X)^{-1}X'H^{-1}ZG = -(X'H^{-1}X)^{-1}X'R^{-1}ZK \]
\[ C^{ZX} = -GZ'H^{-1}X(X'H^{-1}X)^{-1} = KR^{-1}Z'X(X'H^{-1}X)^{-1} \]

and from (2.19)
\[ C^{ZZ} = (Z'R^{-1}Z + G^{-1})^{-1} + GZ'H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1}ZG = K + KR^{-1}Z'X(X'H^{-1}X)^{-1}X'R^{-1}ZK \]

This allows the inverse of $C$ to be expressed as
\[ C^{-1} = \begin{bmatrix} (X'H^{-1}X)^{-1} & -(X'H^{-1}X)^{-1}X'R^{-1}ZK \\ -KR^{-1}Z'X(X'H^{-1}X)^{-1} & K + KR^{-1}Z'X(X'H^{-1}X)^{-1}X'R^{-1}ZK \end{bmatrix} \]
Consider

\[
WC^{-1}W' = \begin{bmatrix} X & Z \end{bmatrix} \begin{bmatrix} (X'X^{-1}X)^{-1} & -(X'X^{-1}X)^{-1}X'R^{-1}ZK \\ KR^{-1}Z'X(X'X^{-1}X)^{-1} & K + KR^{-1}Z'X(X'X^{-1}X)^{-1}X'R^{-1}ZK \end{bmatrix} \begin{bmatrix} X' \\ Z' \end{bmatrix}
\]

\[
= X(X'X^{-1}X)^{-1}X' - ZKR^{-1}Z'X(X'X^{-1}X)^{-1}X' - X(X'X^{-1}X)^{-1}X'R^{-1}ZKZ' + ZKZ' + ZKR^{-1}Z'X(X'X^{-1}X)^{-1}X'R^{-1}ZKZ'
\]

and then

\[
R^{-1} - R^{-1}WC^{-1}W'R^{-1} \\
= R^{-1} - R^{-1}X(X'X^{-1}X)^{-1}X'R^{-1} + R^{-1}ZKR^{-1}Z'X(X'X^{-1}X)^{-1}X'R^{-1} \\
+ R^{-1}X(X'X^{-1}X)^{-1}X'R^{-1}ZKZ'R^{-1} - R^{-1}ZKZ'R^{-1} \\
- R^{-1}ZKR^{-1}Z'X(X'X^{-1}X)^{-1}X'R^{-1}ZKZ'R^{-1} \\
= R^{-1} - R^{-1}ZKZ'R^{-1} - (R^{-1} - R^{-1}ZKZ'R^{-1})X(X'X^{-1}X)^{-1}X'R^{-1} \\
+ (R^{-1} - R^{-1}ZKZ'R^{-1})X(X'X^{-1}X)^{-1}X'R^{-1}ZKZ'R^{-1} \\
= R^{-1} - R^{-1}ZKZ'R^{-1} \\
- (R^{-1} - R^{-1}ZKZ'R^{-1})X(X'X^{-1}X)^{-1}X'(R^{-1} - R^{-1}ZKZ'R^{-1}) \\
= H^{-1} - H^{-1}X(X'X^{-1}X)^{-1}X'H^{-1} \\
= P
\]

Hence the matrix \( P = H^{-1} - H^{-1}X(X'X^{-1}X)^{-1}X'H^{-1} \) can also be written as

\[
P = R^{-1} - R^{-1}WC^{-1}W'R^{-1}
\]

Result A.6

\[
PHP = P
\]

Proof

\[
PHP = H^{-1}(I - X(X'X^{-1}X)^{-1}X'H^{-1})HH^{-1}(I - X(X'X^{-1}X)^{-1}X'H^{-1}) \\
= H^{-1}(I - 2X(X'X^{-1}X)^{-1}X'H^{-1} + X(X'X^{-1}X)^{-1}X'H^{-1}X(X'X^{-1}X)^{-1}X'H^{-1}) \\
= H^{-1}(I - X(X'X^{-1}X)^{-1}X'H^{-1}) \\
= P
\]
Result A.7

\[
\frac{\partial P}{\partial \kappa_i} = -P \dot{H}_i P
\]

Proof

\[
\frac{\partial P}{\partial \kappa_i} = \frac{\partial}{\partial \kappa_i} (H^{-1} - H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1})
= \frac{\partial H^{-1}}{\partial \kappa_i} + \frac{H^{-1}}{\partial \kappa_i} X(X'H^{-1}X)^{-1}X'H^{-1} - H^{-1}X \frac{\partial (X'H^{-1}X)^{-1}}{\partial \kappa_i} X'H^{-1}
\]

\[
= -H^{-1}\dot{H}_i H^{-1} + H^{-1}\dot{H}_i H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1}
- H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1}H_i H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1}
+ H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1}\dot{H}_i H^{-1}
= -H^{-1}\dot{H}_i P + H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1}\dot{H}_i P
= -(H^{-1} - H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1})\dot{H}_i P
= -P \dot{H}_i P
\]

Result A.8

\[SRS = S\]

Proof

First consider \[SR\]

\[
SR = (R^{-1} - R^{-1}X(X'R^{-1}X)^{-1}X'R^{-1})R
= I - R^{-1}X(X'R^{-1}X)X'
\]

and hence

\[
SRS = S - R^{-1}X(X'R^{-1}X)^{-1}X'S
= S - R^{-1}X(X'R^{-1}X)^{-1}X'(R^{-1} - R^{-1}X(X'R^{-1}X)^{-1}X'R^{-1})
= S - R^{-1}X(X'R^{-1}X)^{-1}X'R^{-1}
\]

\[
+ R^{-1}X(X'R^{-1}X)^{-1}X'R^{-1}X(X'R^{-1}X)^{-1}X'R^{-1}
= S - R^{-1}X(X'R^{-1}X)^{-1}X'R^{-1} + R^{-1}X(X'R^{-1}X)^{-1}X'R^{-1}
= S
\]
Result A.9

\[ C^{Z} = (G^{-1} + \Omega)^{-1} \]

Proof

Using Result A.4, if the matrices \( X \) and \( V \) are both of full column rank and satisfy
\( V'X = 0 \), and \( H^{-1} \) and \( R^{-1} \) are both positive definite, then
\[
P = H^{-1} - H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1} = V(V'H)^{-1}V'
\]
and
\[
S = R^{-1} - R^{-1}X(X'R^{-1}X)^{-1}X'R^{-1} = V(V'RV)^{-1}V'
\]

We now consider
\[
P = H^{-1} - H^{-1}X(X'H^{-1}X)^{-1}X'H^{-1} = V(V'H)^{-1}V'
\]
\[
= V(V'(ZGZ' + R)V)^{-1}V'
\]
\[
= V(V'(ZGZ'V + V'RV)^{-1}V'
\]
\[
= V \left( (V'RV)^{-1} - (V'RV)^{-1}V'Z (G^{-1} + Z'V(V'RV)^{-1}V'Z)^{-1} Z'V(V'RV)^{-1} \right) V' \quad \text{(Using Result A.2)}
\]
\[
= V(V'RV)^{-1}V' - V(V'RV)^{-1}V'Z (G^{-1} + Z'V(V'RV)^{-1}V'Z)^{-1} Z'V(V'RV)^{-1}V'
\]
\[
= S - SZG^{-1} + Z'SZ^{-1} Z'S \quad \text{(Using Result A.2)}
\]

Hence
\[
C^{Z} = G - GZ'PZG
\]
\[
= G - GZ'(SZ^{-1} + ZGZ')^{-1} ZG \quad \text{(Using Result A.2)}
\]
\[
= (G^{-1} + Z'SZ)^{-1} \quad \text{(Using Result A.2)}
\]
\[
= (G^{-1} + \Omega)^{-1}
\]
Result A.10
If $y$ is such that $E[y] = \mu$ and $\text{Var}[y] = \Sigma$, then for any symmetric matrix $D$

$$E[y'Dy] = \text{tr}(\Sigma D) + \mu'D\mu$$

Result A.11
If $y_1$ and $y_2$ are vectors with a joint distribution given by

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \sim N\left(\begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}\right)$$

then the conditional distribution of $y_1$ given $y_2$ is given by

$$y_1|y_2 \sim N(\mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(y_2 - \mu_2), \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21})$$

and the conditional distribution of $y_2$ given $y_1$ is

$$y_2|y_1 \sim N(\mu_2 + \Sigma_{21}\Sigma_{11}^{-1}(y_1 - \mu_1), \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12})$$

The expectation of $y_1y_1'$ is given by

$$E[y_1y_1'] = \mu_1\mu_1' + \Sigma_{11}$$

and the expectation of $y_1y_2'$ is given by

$$E[y_1y_2'] = \mu_1\mu_2' + \Sigma_{12}$$

Result A.12
If

$$C = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

then the inverse of $C$ is given by

$$C^{-1} = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}^{-1}$$

$$= \begin{bmatrix} (C_{11} - C_{12}C_{22}^{-1}C_{21})^{-1} & -C_{11} + C_{12}C_{22}^{-1}C_{21})^{-1}C_{12}C_{22}^{-1} \\ -C_{22}^{-1}C_{21}(C_{11} - C_{12}C_{22}^{-1}C_{21})^{-1} & C_{22}^{-1} + C_{22}^{-1}C_{21}(C_{11} - C_{12}C_{22}^{-1}C_{21})^{-1}C_{12}C_{22}^{-1} \end{bmatrix}$$

$$= \begin{bmatrix} C_{11}^{-1} & -C_{11}C_{12}C_{22}^{-1} \\ -C_{22}^{-1}C_{21}C_{11}^{-1} & C_{22}^{-1} + C_{22}^{-1}C_{21}C_{11}^{-1}C_{12}C_{22}^{-1} \end{bmatrix}$$

Proof
See Mardia et al. (1979)
Result A.13
If $A$ is a $m \times m$ matrix and $B$ is a $n \times n$ matrix then
\[
(A \otimes B)^{-1} = A^{-1} \otimes B^{-1}
\]
\[
|A \otimes B| = |A|^n |B|^m
\]
\[
\text{tr}(A \otimes B) = \text{tr}(A)\text{tr}(B)
\]

If $A$ is an $m \times n$ matrix, $B$ is an $n \times n$ matrix and $C$ is an $m \times m$ matrix then
\[
(\text{vec}(A))'(B \otimes C)\text{vec}(A) = \text{tr}(B'A'CA)
\]

If $A$ is an $m \times n$ matrix and $B$ is an $n \times p$ matrix
\[
\text{Vec}(AB) = (I_p \otimes A)\text{Vec}(B)
\]

Proof
See Harville (2000)

Result A.14
For any matrix $A$
\[
r(A^\top A) = r(AA^\top) = r(A)
\]
\[
r(A'A) = r(A)
\]
\[
r(A) = \text{tr}(A) \text{ if } A \text{ is idempotent}
\]

Proof
See Harville (2000)

Result A.15
\[
\text{tr}(W(W'W)^{-1}W') = r(W)
\]

Proof
Using Result A.14
\[
\text{tr}(W(W'W)^{-1}W') = \text{tr}(W'W(W'W)^{-1})
\]
\[
= r(W'W(W'W)^{-1})
\]
\[
= r(W'W)
\]
\[
= r(W)
\]
Result A.16
If $A = \{a_{ij}\}$ is an $m \times n$ matrix, $B = \{b_{ij}\}$ is an $n \times p$ matrix and $C = \{c_{ij}\}$ is an $p \times q$ matrix then

$$AB = \{\sum_{k=1}^{n} a_{ik}b_{kj}\}$$

and hence

$$ABC = \{\sum_{l=1}^{p} \sum_{k=1}^{n} a_{ik}b_{kl}c_{lj}\}$$

Result A.17
If $A$ and $B$ are $n \times n$ matrices that are partitioned into $p^2$ submatrices each of size $q \times q$ then

$$AB = \{\sum_{k=1}^{p} A_{ik}B_{kj}\}$$

and

$$\text{tr}(AB) = \sum_{l=1}^{p} \sum_{k=1}^{p} \text{tr}(A_{lk}B_{kl})$$

Result A.18
If $M$ is a $3 \times 3$ matrix given by

$$M = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

then the inverse of this matrix is given by

$$M^{-1} = \frac{1}{a(ei - hf) + b(fg - di) + c(dh - eg)} \begin{bmatrix} ei - fh & ch - bi & bf - ce \\ fg - di & ai - cg & cd - af \\ dh - eg & bg - ah & ae - bd \end{bmatrix}$$
Result A.19

From (6.8) in Chapter 3, the AI update of $\kappa$ is given by

$$
\kappa^{(m+1)} = \kappa^{(m)} + \mathcal{I}_A(\kappa^{(m)}, \kappa^{(m)})^{-1} U(\kappa^{(m)})
$$

where

$$
\delta^{(m)} = \mathcal{I}_A(\kappa^{(m)}, \kappa^{(m)})^{-1} U(\kappa^{(m)})
$$

and hence

$$
\mathcal{I}_A(\kappa^{(m)}, \kappa^{(m)})\delta^{(m)} = U(\kappa^{(m)})
$$

We partition $\kappa^{(m)}$ into $\kappa^{(m)} = \left[ \begin{array}{c} \kappa_1^{(m)} \\ \kappa_2^{(m)} \end{array} \right]$, and hence $U(\kappa^{(m)}) = \left[ \begin{array}{c} U(\kappa_1^{(m)}) \\ U(\kappa_2^{(m)}) \end{array} \right]$, $\mathcal{I}_A(\kappa^{(m)}, \kappa^{(m)}) = \left[ \begin{array}{cc} \mathcal{I}_A(\kappa_1^{(m)}, \kappa_1^{(m)}) & \mathcal{I}_A(\kappa_1^{(m)}, \kappa_2^{(m)}) \\ \mathcal{I}_A(\kappa_2^{(m)}, \kappa_1^{(m)}) & \mathcal{I}_A(\kappa_2^{(m)}, \kappa_2^{(m)}) \end{array} \right]$ and $\delta^{(m)} = \left[ \begin{array}{c} \delta_1^{(m)} \\ \delta_2^{(m)} \end{array} \right]$. Then we have

$$
\left[ \begin{array}{cc} \mathcal{I}_A(\kappa_1^{(m)}, \kappa_1^{(m)}) & \mathcal{I}_A(\kappa_1^{(m)}, \kappa_2^{(m)}) \\ \mathcal{I}_A(\kappa_2^{(m)}, \kappa_1^{(m)}) & \mathcal{I}_A(\kappa_2^{(m)}, \kappa_2^{(m)}) \end{array} \right] \left[ \begin{array}{c} \delta_1^{(m)} \\ \delta_2^{(m)} \end{array} \right] = \left[ \begin{array}{c} U(\kappa_1^{(m)}) \\ U(\kappa_2^{(m)}) \end{array} \right]
$$

Solving for $\delta_2^{(m)}$ gives

$$
\delta_2^{(m)} = \mathcal{I}_A(\kappa_2^{(m)}, \kappa_2^{(m)})^{-1} \left( U(\kappa_2^{(m)}) - \mathcal{I}_A(\kappa_2^{(m)}, \kappa_1^{(m)})\delta_1^{(m)} \right)
$$

If we constrain the update for $\kappa_1^{(m+1)} = \kappa_1^{(m+1)*}$, then $\delta_1^{(m)*} = \kappa^{(m+1)*} - \kappa^{(m)}$, and we can adjust the update for $\kappa_2^{(m+1)}$ to be $\kappa_2^{(m+1)*} = \kappa_2^{(m)} + \delta_2^{(m)*}$ where

$$
\delta_2^{(m)*} = \mathcal{I}_A(\kappa_2^{(m)}, \kappa_2^{(m)})^{-1} \left( U(\kappa_2^{(m)}) - \mathcal{I}_A(\kappa_2^{(m)}, \kappa_1^{(m)})\delta_1^{(m)*} \right)
$$
Bibliography


SPSS Inc. (2002). *Linear mixed-effects modeling in SPSS*. SPSS Inc.


