Flows Through Helical Pipes with Elliptic Cross-Sections

Jordan Belperio

June 5, 2018

Thesis submitted for the degree of
Masters of Philosophy
in
Applied Mathematics
at The University of Adelaide
Faculty of Engineering, Computer and Mathematical Sciences
School of Mathematical Sciences

THE UNIVERSITY
of ADELAIDE

Contents

List of Tables

List of Figures

Signed Statement

I certify that this work contains no material which has been accepted for the award of any other degree of diploma in my name, 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 in the text. In addition, I certify that no part of this work will, in the future, be used in submissions in my name, for any other degree or diploma in any university or other tertiary institution without the prior approval of the University of Adelaide and where applicable, any partner institution responsible for the joint award of this degree.

I give 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.

I also give permission for the digital version of my thesis to be made available on the web, via the University's digital research repository, the Library Search and also through web search engines, unless permission has been granted by the University to restrict access for a period of time.

I acknowledge the support I have received for my research through the provision of an Australian Government Research Training Program Scholarship.

Signed:

Date: 17/5/2018

Acknowledgements

I wish to acknowledge those who have made this thesis possible. Firstly, to Dr. Trent Mattner, for his seemingly unending wisdom and patience. To Prof. Jim Denier, and Prof. Yee Khong for their insight into new areas of research and enthusiasm for learning. To Dr. David Wilke, for his previous work in modeling umbilical cords, and for his personal correspondence and help with `oomph-lib`. To Prof. Matthias Hiel and Dr. Andrew Hazel for developing `oomph-lib`. Finally, to my family and many friends for their care and support.

Dedication

For my family, who never stopped believing in me.

Abstract

Flows through helical pipes are important in many applications, such as blood flow through the coiled veins and arteries of an umbilical cord, and flow through industrial heat exchangers and reactors. This research examines flows through helical pipes with elliptic cross-section. The incompressible Navier-Stokes equations are solved for steady and unsteady flows through these geometries at low Reynolds numbers using the finite-element method library `oomph-lib`. The effect of changes in the various non-dimensional parameters such as the aspect ratio of the ellipse, curvature and torsion of the helix and the Reynolds number on the flow dynamics are explored. The computed results are compared with theoretical solutions, previous computations and experimental results to verify and validate the solver.

Trends for elliptical cross-sections are usually similar to those of circular cross-sections. However for a geometry of very high torsion and low aspect ratio, the flow develops a persistent oscillation with streamwise distance along the pipe. This behaviour is not present for circular cross-sections. The aspect ratio controls the period of the oscillation and the Reynolds number controls the magnitude of the oscillation. At low Reynolds and Strouhal number, steady results can be used to predict time-averaged pressure values of unsteady flows. This is important for umbilical cords as it allows for simple predictions of pressure drops using empirical data.

Chapter 1

Introduction

1.1 Background

Flows through helical pipes occur both in nature and in industry. A recent area of interest of naturally occurring helical flows is flow through umbilical cords. The umbilical cord is the connecting cord from a placenta to a developing embryo or foetus. The cord supplies oxygen and nutrients through a single vein, and deoxygenated blood from the foetus returns through two arteries (Cunningham et al. 2009). Ghezzi et al. (2001) demonstrated that the cross-section of the vein and arteries are elliptical. These three blood vessels reside in a gelatinous substance known as Wharton's Jelly, as seen in figure 1.1. Sonographic measurements were in a plane close to the cord insertion at the foetal abdomen to create the image.

The cord and blood vessels are usually coiled. An umbilical coil is defined as one complete spiral of 360° of the umbilical cord, including the vessels, around itself. In clinical studies, coiling is often quantified by the umbilical coiling index (UCI)

$$\text{UCI} = \frac{\text{Number of coils}}{\text{Length of cord (cm)}}. \tag{1.1}$$

Abnormal UCI is correlated with adverse pregnancy outcomes such as intrauterine death, preterm delivery, intrapartum fetal heart rate decelerations, operative delivery for metal distress, meconium staining and anatomical-karyotypic abnormalities (de Laat et al. 2005). However, the UCI has a number of limitations, as it does not account for non-uniformly distributed coils or the existence of knots (Khong 2010). Umbilical cords with an UCI $\leq 10^{\text{th}}$ percentile are considered *hypocoiled*, and exhbit undercoiling. Conversely, umbilical cords with an UCI $\geq 90^{\text{th}}$ percentile exhibit overcoiling and are

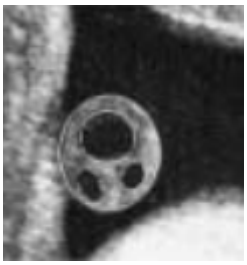


Figure 1.1: Cross-section of umbilical cord. The grey circle is the Wharton's jelly, the large black ellipse is the vein and the two smaller ellipses are the arteries (Ghezzi et al. 2001).

considered *hypercoiled* (de Laat et al. 2005). Cords that are not hypercoiled or hypocoiled are *normocoiled*.

In an effort to better understand blood flow through the umbilical cord and the role of the UCI, Wilke (2016) modelled the veins and arteries as helices of varying pitch and helical radius with circular cross-section and non-compliant walls. He found that the fluid flow and pressure drop through the helix does not depend on the UCI alone and proposed an alternative index based on the pressure drop through the helix. In reality, umbilical cords have compliant walls that may not be exactly circular in cross section, especially when the cord is highly deformed.

Flow through umbilical arteries is pulsatile. Weerappuli & Parakrama (1987) simulated Newtonian, pulsatile flow in arteries using the finite-element method to provide a computational method to investigate arterial flow. This was to provide a non-invasive method of finding changes in arterial flows, as cardiovascular diseases represent a major cause of death. They found that a finite-element method could satisfactorily predict flow and pressure propagation along segments of arteries. It was also found that the effects of vessel compliance, and hence, the effects of wall structure interactions was significant in determining the flow.

In industry, helical pipes are used as motionless mixers, heat exchangers and reactors. Helical pipes are often used to fit large pipe lengths into small physical spaces; Vashisth et al. (2008) notes that curved and helical pipes have inherent advantages over straight pipes for these processes. The advantages include enhanced cross-sectional mixing, reduction in axial dispersion and better heat and mass transfer.

1.2 Modelling incompressible Newtonian flow

This research will only consider Newtonian flows with constant density that can be described by the Navier-Stokes equations,

$$\hat{\rho}\left(\frac{\partial \hat{u}_i}{\partial \hat{t}} + \hat{u}_j \frac{\partial \hat{u}_i}{\partial \hat{x}_j}\right) = -\frac{\partial \hat{P}}{\partial \hat{x}_i} + \frac{\partial}{\partial \hat{x}_j}\left[\hat{\mu}\left(\frac{\partial \hat{u}_i}{\partial \hat{x}_j} + \frac{\partial \hat{u}_j}{\partial \hat{x}_i}\right)\right]. \tag{1.2}$$

Blood flow is two phase in nature, consisting of red blood cells suspended in plasma. Blood flow is assumed to be incompressible and Newtonian to simplify the governing equations. In equation (1.2), $\hat{x}_i$ are the Cartesian coordinates $(\hat{x}, \hat{y}, \hat{z})$, $\hat{u}_i$ is the velocity in the direction $\hat{x}_i$, $\hat{\mu}$ is the dynamic viscosity of the fluid, $\hat{\rho}$ is the fluid density, $\hat{t}$ is the time and $\hat{P}$ is the pressure. The hat denotes dimensional quantities. These equations form a set of non-linear partial differential equations and solving them, if possible, will determine the velocity field and pressure for given boundary conditions. By using problem-specific reference quantities $\hat{U}$, the bulk velocity, $\hat{D}$, the effective diameter and $\hat{T}$, the period of pulsation, the dimensional equations in (1.2) are non-dimensionalised. The non-dimensional Navier-Stokes equations

$$St\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} = -\frac{\partial P}{\partial x_i} + \frac{1}{Re}\frac{\partial}{\partial x_j}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) \tag{1.3}$$

where Re is the Reynolds number and St is the Strouhal number. These non-dimensional numbers are

$$Re = \frac{\hat{U}\hat{D}\hat{\rho}}{\hat{\mu}}, \quad St = \frac{\hat{D}}{\hat{U}\hat{T}}.$$

The Reynolds number is the ratio of inertial forces to viscous forces. High Reynolds-number flows tend to be more susceptible to instabilities than low Reynolds-number flows. The Reynolds number can be used to predict turbulent transition. In a straight pipe, a flow with Re less than approximately 2000 can be considered laminar or stable, while flows with a Re of over 4000 are often turbulent (Ward-Smith 1980).

In this work, the Strouhal number is used as a dimensionless frequency for periodic flows. At high Strouhal numbers, $St \approx 1$, the flow is dominated by viscous forces, resulting in plug shaped velocity profiles. At low Strouhal numbers, $St \approx 10^{-4}$, the flow is dominated by the steady component of the velocity.

Along with the equation for the continuity of an incompressible fluid

$$\frac{\partial u_i}{\partial x_i} = 0,$$

equation (1.3) forms a system of four partial differential equations in the four unknowns, P, u_1, u_2 and u_3. The above system only has analytical solutions for a few simplified problems. For more difficult problems, like turbulent flow and flow in complex geometries, computational methods like the finite-element method can find very accurate approximate solutions.

1.2.1 Modelling pulsatile flow

Blood flow in a human body exhibits a pulsatile waveform which is dependent on the pressure pulse generated by the heart. Flows of this nature in an infinitely long straight, circular pipe are periodic. Womersley (1955) found, using exact methods, that the velocity is

$$W(r,t) = 2W_{b,s}\left[1-\left(\frac{r}{R}\right)^2\right] + \sum_{n=1}^{N} \mathrm{Re}\left[\frac{P_n i}{\rho\omega_n}\left(\frac{J_0(i^{\frac{3}{2}}\alpha_n\frac{r}{R})}{J_0(i^{\frac{3}{2}}\alpha_n)}-1\right)e^{i\omega_n t}\right], \quad (1.4)$$

where R is the pipe radius, $W_{b,s}$ is the bulk velocity of the steady component of the flow, Re denotes the real component, $\omega_n = 2\pi n/T$ is the circular frequency of the n-th harmonic, $i = \sqrt{-1}$, P_n are the complex pressure coefficients, $\alpha = (2\pi ReSt)^{\frac{1}{2}}$ is the Womersley number and J_0 is the zeroth order Bessel function. The pressure coefficients are determined by choosing a cross-sectionally averaged velocity or centreline velocity.

A wide variety of pulsatile waveforms are observed in arteries. Sherwin & Blackburn (2005) used a simple cross-sectionally averaged velocity profile of

$$\bar{W}(t) = \langle\bar{W}\rangle\left[1 + a_1\sin(\omega_1 t) + a_2\cos(\omega_2 t)\right], \quad (1.5)$$

where

$$\langle\bar{W}\rangle = \frac{1}{T}\int_0^T \bar{W}(t)\,\mathrm{d}t$$

is the time and cross-sectionally averaged velocity and

$$\bar{W}(t) = \frac{2}{R^2}\int_0^R W(r,t)\,r\,\mathrm{d}r$$

is the cross-sectionally averaged velocity, to investigate blood flow through a stenosed artery. The investigation by Sherwin & Blackburn (2005) considered flows of Reynolds number ranging from 250 to 550, and the Strouhal numbers ranged from 0.4 to 0.13. The flow profile in equation (1.5) is used to verify the unsteady solver in section 2.7.

1.3 Helical pipes and elliptic cross-sections

1.3.1 The parameterisation of helical pipes

Wang (1981) introduced a non-orthogonal coordinate system using the Frenet triad of unit vectors for a spatial curve in order to study the complex secondary flows found in helical pipes. Wang constructed a coordinate system (s, r, θ) where s is the arc-length of the centreline, r is the distance from the centre of the pipe, and θ is the azimuthal coordinate. This coordinate system is shown in figure 1.2. Any Cartesian position vector $\mathbf{x}$ can be expressed as

$$\mathbf{x} = \mathbf{R}(s) + r\cos(\theta)\mathbf{N}(s) + r\sin(\theta)\mathbf{B}(s) \tag{1.6}$$

where

$$\mathbf{R}(s) = \hat{R}_c \cos\left(\frac{s}{\sqrt{\hat{R}_c^2 + \hat{H}^2}}\right)\mathbf{i} + \hat{R}_c \sin\left(\frac{s}{\sqrt{\hat{R}_c^2 + \hat{H}^2}}\right)\mathbf{j} + \frac{\hat{H}s}{\sqrt{\hat{R}_c^2 + \hat{H}^2}}\mathbf{k}$$

is the pipe centreline, $\hat{R}_c$ is the radius of the helix, $2\pi\hat{H}$ is the vertical displacement between two coils and a hat denotes a dimensional quantity.

To determine $\mathbf{N}$, the unit normal, $\mathbf{B}$, the unit binormal, $\hat{\kappa}$, which is the dimensional curvature of the centreline, and $\hat{\tau}$, which is the dimensional torsion of the centreline, the relations

$$\mathbf{T} = \frac{d\mathbf{R}}{ds}, \quad \mathbf{N} = \frac{1}{\hat{\kappa}}\frac{d\mathbf{T}}{ds}, \quad \mathbf{B} = \mathbf{T} \times \mathbf{N}, \tag{1.7}$$

and the Frenet formulae

$$\frac{d\mathbf{N}}{ds} = \hat{\tau}\mathbf{B} - \kappa\mathbf{T}, \quad \frac{d\mathbf{B}}{ds} = -\hat{\tau}\mathbf{N} \tag{1.8}$$

are used. The dimensional curvature and torsion are

$$\hat{\kappa} = \frac{\hat{R}_c}{\hat{R}_c{}^2 + \hat{H}^2} \quad \text{and} \quad \hat{\tau} = \frac{\hat{H}}{\hat{R}_c{}^2 + \hat{H}^2}. \tag{1.9}$$

The curvature and torsion of the pipe centreline have a significant effect on the cross-flow in the **N**-**B** plane. The coordinate system introduced by Wang (1981) is used in this thesis to define the pipe in Cartesian space. The finite-element solver `oomph-lib` solves the problem in Cartesian x, y, z coordinates, so it is not necessary to transform the governing equations into Wang's coordinate system.

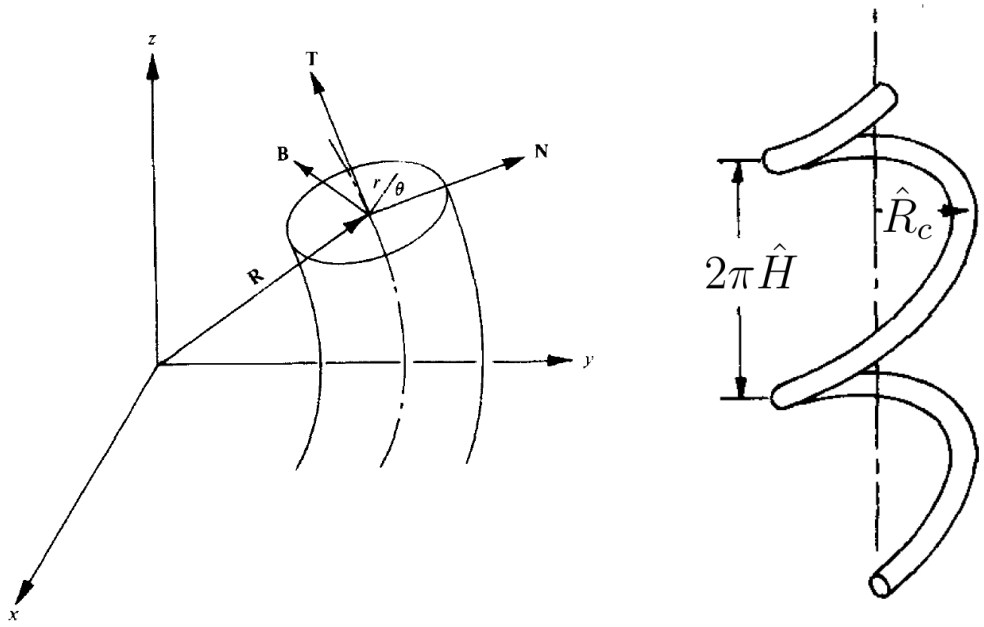


Figure 1.2: Coordinate system used by Wang (1981).

1.3.2 Flows trough helical pipes

The analysis of flows in curved pipes began with Dean & Hurst (1927) and Dean (1928). Dean used analytical methods and assumed that the cross-flow motion is approximately a uniform stream near the centre of the pipe. The cross-flow motion decreases the rate of flow produced by a given pressure gradient compared to a straight pipe. The cross-flow motion also causes the location of the maximum magnitude of axial velocity to shift to the outside wall. The cross-flow motion is indicated by Dean to be two counter-rotating vortices, with the flow moving to the outside wall in the middle and the inside wall at the top and bottom. The vortices are shown in figure 1.3.

Flows through helical pipes have been a research interest since the early 1980s, with Wang (1981) and Germano (1982) amongst the first to attempt to define coordinate systems and to deduce the effect on the curvature and torsion on the flow in a helical pipe. Both studies utilised perturbation methods to investigate gently curving helical pipes by using the product of pipe radius ($\hat{D}/2$) and dimensional curvature ($\hat{\kappa}$) as the small non-dimensional parameter. Germano found that the torsion produced a second order effect on the secondary flow, but Wang found that the torsion produced a first order effect. Later, Tuttle (1990) reconciled the conflicting works of Germano and

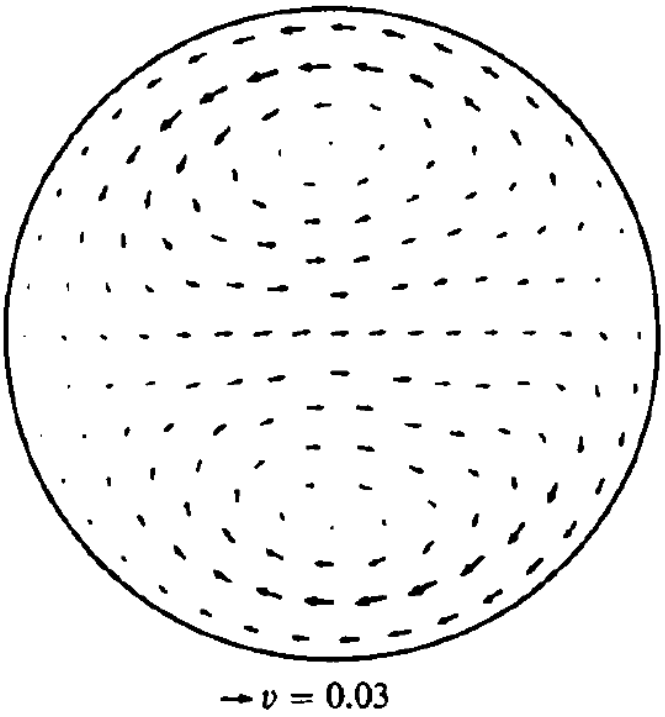


Figure 1.3: Dean vortices in flow through a curved pipe (Liu & Masliyah 1993). The outside wall is on the right of the page.

Wang by using a combination of both coordinate systems. Tuttle found that the effect of torsion, is in fact, first order. Small changes in the torsion of the pipe centreline lead to large distortions of the two recirculating vortices. At moderate pitch, the two vortices coalesce into a single vortex. Manlapaz & Churchill (1980) were amongst the first to attempt to find a numerical solution to fully developed laminar flow in a helically coiled tube of finite pitch. They found that the flow through this geometry was intermediate between flow for a straight pipe and a curved pipe.

There has been some recent works for helical pipes with cross-sectional geometries that are not circular or elliptical, including rectangles and annuli. Nobari & Malvandi (2013) considered flow through a helical annulus. By using the Frenet basis vectors and a coordinate system very similar to Germano (1982), Nobari & Malvandi (2013) developed a second order finite-difference method to solve the governing equations. They found that helical annuli with small torsion acted very similar to curved pipes. As torsion increased, the two Dean vortices would merge to form a single vortex in the secondary flow and the location of maximum axial velocity would shift to the centreline.

Morales & Rosa (2012) used a finite-volume method to investigate flows through helical pipes with square and rectangular cross-sections. The research used non-dimensional curvature $\hat{\kappa}\hat{d}_h$, torsion $\hat{\tau}\hat{d}_h$ and Re, where $\hat{d}_h$ is the hydraulic diameter, to characterise the flow. For $\hat{\kappa}\hat{d}_h = 0.471$, $\hat{\tau}\hat{d}_h = 0.187$, and $Re = 144$, which are moderate values, the location of maximum axial velocity shifts to the outside wall and two counter-rotating vortices form. For flow at $Re = 501$ in the same pipe, the shift to the outside wall is more prominent. The secondary flow is towards the outside wall in the middle of the pipe, and towards the inside wall at the top and bottom of the pipe at both Reynolds numbers.

Liu & Masliyah (1993) used a numerical method to investigate the effect of curvature and torsion on the axial flow, cross-flow and pressure gradient through helical pipes for a range of Reynolds numbers. An analysis in the limit of small dimensionless parameters $\lambda = \hat{\kappa}\hat{D}/2$ and $\eta = \hat{\tau}\hat{D}/2$ led to two dominant parameters, the Germano number $Gn = Re\,\eta$ and the Dean number $Dn = Re\sqrt{\lambda}$. Consideration of the cross-flow led to a third dimensionless group $\gamma = Gn/(Dn)^{3/2}$. For a flow of constant curvature and Reynolds number, they found that increasing the torsion from zero lead to a change in location of the maximum magnitude of axial flow. As torsion was increased from zero, the location of the maximum magnitude of axial velocity shifts from the outside wall, to the upper inside wall and then the axial flow approaches a Poiseuille-type flow. For increasing γ, the secondary flow changes from two vortices similar to curved pipe flow, to one large vortex, to two vortices positioned near the inside and outside walls of the pipe. The flow in the middle of the pipe is towards the top, and the flow is towards the bottom of the pipe at the inside and outside walls.

Selvarasu & Tafti (2012) investigated the effect of motion of the coronary artery by considering dynamic changes in the curvature and torsion of helical pipes. They used computational fluid dynamics simulations of pulsatile flows through helical geometries with dynamically changing boundaries. Significant changes to the wall shear stress were found under dynamic torsion and curvature. It was found that, for a dynamic change in torsion of 15% the wall shear stress would change by 3%, for a dynamic change in curvature of 20% the wall shear stress would change by 22% and for a dynamic change in both curvature and torsion of 43% and 17% respectively, the change in wall shear stress would be 26% compared with flow through pipes without dynamic changes.

Wilke (2016) modelled flows through helical pipes with circular cross-sections to investigate the efficacy of the UCI in predicting blood vessel pressure drop. Wilke used the software package `oomph-lib` to find finite-element solutions of steady and unsteady flows through these models. The centrelines

of the umbilical models used in this study are taken from Wilke (2016), and are given elliptical cross-sections to examine the effect of the shape of the cross-section on the flow.

1.3.3 Flows through pipes with elliptic cross-sections

Fully developed steady flow through a straight pipe with an elliptic cross-section has an exact solution. This solution is presented in Lekner (2007). The velocity profile for this solution is

$$u_{\text{axial}} = \frac{2Q}{\pi ab}\left(1 - \frac{x^2}{a^2} - \frac{y^2}{b^2}\right), \tag{1.10}$$

where u_{axial} is the axial velocity, x and y are the pipe cross-section coordinates, a and b are the x and y axes of the cross-section respectively and Q is the volumetric flow rate. The volumetric flow rate is given as a function of the pressure gradient by

$$Q = \frac{\pi}{4\mu}\frac{\partial P}{\partial z}\frac{(ab)^3}{a^2 + b^2}, \tag{1.11}$$

where z is the axial coordinate.

Germano (1989) used the same methods as Germano (1982) to find an asymptotic solution to steady flow through helical pipes with elliptic cross-sections. Through this method, Germano (1989) found explicit formulae for the Cartesian components of the cross-flow in the limit of small curvature and Reynolds number. This paper provides the basis for the numerical verification of the steady solver found in section 2.6.

Recently, Levy et al. (2015) used experimental methods to investigate flows through helical pipes by coiling a straight plastic tube. The coiling of one of the experimental pipes lead to the deformation of the cross-section into a shape close to an ellipse. Time Resolved Particle Image Velocimetry was used to measure flow within the coil. Experiments were performed for steady flows and flows that are suddenly stopped after reaching a steady state. They found that the vorticity field and the component of velocity in the direction of the helical axis follow the same trends for both circular cross-sections and elliptic cross-sections when the flow is steady. It was also found that, in the case where the flow is suddenly stopped, the cross-flow becomes weaker as time passes. The experimental pipes are modelled in this thesis to compare the experimental results with simulations.

1.4 Aims

The aim of this project is to investigate the flow through a helical pipe with an elliptical cross section. In particular, the objectives are to:

1. Develop and verify a `c++` code that uses the `oomph-lib` library to find finite-element solutions of steady and unsteady flow through helical pipes of elliptic cross section.

2. Compare computational results with the experimental results in Levy et al. (2015).

3. Investigate the effect of the aspect ratio and orientation of the cross section on steady and pulsatile flow through typical models of umbilical cord vessels.

Chapter 2 presents the mathematical model of the flows, the finite-element method, and its implementation (Aim 1). The solver is verified for unsteady and steady cases. The steady verification considers an asymptotic solution to flow through helical pipes with elliptic cross-sections given by Germano (1989). The unsteady verification considers the exact solution to pulsatile flow through a straight pipe with a circular cross-section provided by Sherwin & Blackburn (2005). The finite-element method is implemented using `oomph-lib`, an open-source library in `c++`, available from the University of Manchester (Heil & Hazel 2017).

In Chapter 3, results from the computational model are compared with experimental results obtained by Levy et al. (2015) (Aim 2). Cross-flow velocities and axial vorticity are compared for steady flow and transient stopping flow. The simulations match the helical geometries of the experiment and approximately matches the cross-section of the pipe, but the inlet, outlet and overall length are different. Nevertheless, it is found that the flow becomes fully developed in the helical section of the pipe, so it is expected that the simulated results in that region should be comparable to the experiment. When allowances are made for potential experimental errors, such as misalignment of the measurement plane with the helix, it is found that the flow patterns obtained from the simulations show good qualitative agreement with those of the experiments. However, there are significant quantitative differences between the magnitude of the cross flow velocities and vorticity.

In Chapter 4, the results of changing the aspect ratio for three flow models typical of umbilical cords taken from Wilke (2016) are presented (Aim 3). Of the three models, one is hypocoiled, one is normocoiled and one is hypercoiled. As the coiling increases, the cross-flow changes from

two counter-rotating Dean vortices for the hypocoiled case, to two unequal counter-rotating vortices for the normocoiled case, to a single dominant vortex for the hypercoiled case. The cross-flow pattern is distorted by the change in cross-sectional shape, but not fundamentally changed. For the hypocoiled and normocoiled cords, the location of the maximum magnitude of axial flow shifts away from the helical centre, towards the outside wall. The location of the maximum magnitude of axial flow shifts towards the helical centre for the hypercoiled case, as the torsion of the cord is very high. The axial flow is distorted by the change in cross-sectional shape, but similar to the cross-flow, the axial flow is not fundamentally changed. Remarkably, the flow develops a steady streamwise oscillation when the major axis of the ellipse is aligned with the axis of the helix and the Reynolds number and aspect ratio are sufficiently large. The oscillation is evident in the axial flow, cross-flow and pressure gradient. It was found that, at the dynamic parameters typical of umbilical flow, time-averaged unsteady results compare favourably with steady results for most cases.

Finally, Chapter 5 presents a summary of the results and a discussion of future work that will extend the understanding of these flows. The case of the steady streamwise oscillation should be investigated to determine the values of the geometric and dynamic parameters for which this behaviour occurs. The effect of this behaviour on the flow features should also be investigated. Fluid-structure interactions may be modelled to understand the effect of the compliance of the arteries and veins of the umbilical cord.

Chapter 2

Mathematical model

In most cases, the Navier-Stokes equations in three-dimensions cannot be solved by analytical methods. Instead, numerical methods are often employed to solve difficult fluid mechanics problems. Previous analytical solutions for helical pipe flows, such as Wang (1981), Germano (1982) and Germano (1989) have used perturbation methods to find asymptotic solutions to helical pipe flows. These solutions are only valid for pipes with small curvature and low Reynolds number. A numerical method is required when flows have high Reynolds numbers, significant non-linear effects and complex geometries. The curvature and Reynolds number needed to model umbilical cords are outside the feasible range of asymptotic solutions.

To investigate flows at higher Reynolds numbers and pipes with large curvature, the finite-element method is chosen to solve the Navier-Stokes equations. This method is chosen because complex geometries can be generated with relative ease. For this research, `oomph-lib`, the object-oriented multiphysics library (Heil & Hazel 2017), an open-source library of finite-elements designed for biological flows, is used. `oomph-lib` is chosen for its native ability to adaptively refine meshes, and the support for parallelisation. There are many examples of `oomph-lib` solvers, including one for helical pipes, and the software is open source and free. The following chapter lists the assumptions and methods used in the research, as well as showing how geometries are constructed. The chapter also includes a verification of the steady solver and of the unsteady solver.

2.1 Assumptions

Assumptions are made to ensure that the problem of interest can be solved computationally in a reasonable time, and that the solution found is mean-

ingful in the context of umbilical cords.

- There are no external forces,
- All pipes are rigid and non-deforming,
- The flow is incompressible and Newtonian.

2.2 Dimensional scaling

The dimensional length scale is the pipe diameter, $\hat{D}$. An ellipse does not have a diameter, so $2\sqrt{\hat{a}\hat{b}}$, where $\hat{a}$ is the normal axis of the ellipse and $\hat{b}$ is the binormal axis of the ellipse, is used as the equivalent diameter. This choice of equivalent diameter is used so the area of the elliptic cross-section is equal to the area of the circular cross-section.

The velocity reference scale, $\mathcal{U}$, is based on the volumetric flow rate, $\hat{Q}$,

$$\mathcal{U} = \frac{4\hat{Q}}{\pi\hat{D}^2} = \hat{U}$$

For unsteady flows, the time scale chosen is the period of the fetal heart-beat, $\hat{T}$

$$\mathcal{T} = \hat{T}.$$

A viscous scale is used to non-dimensionalise the pressure drop through the pipe ΔP. This removes the contribution of Re from the pressure drop so that any increase comes from an increase in pipe length or changes in the axial and cross-flow,

$$\Delta P = \frac{\Delta\hat{P}Re\hat{D}^4\pi^2}{16\hat{Q}^2\rho},$$

where Re is the Reynolds number,

$$Re = \frac{\mathcal{U}\mathcal{L}}{\hat{\nu}} = \frac{\hat{U}\hat{D}}{\hat{\nu}},$$

and $\hat{\nu}$ is the kinematic viscosity of the flow medium. For unsteady flows, the Strouhal number is

$$St = \frac{\mathcal{L}}{\mathcal{U}\mathcal{T}} = \frac{\hat{D}}{\hat{U}\hat{T}}.$$

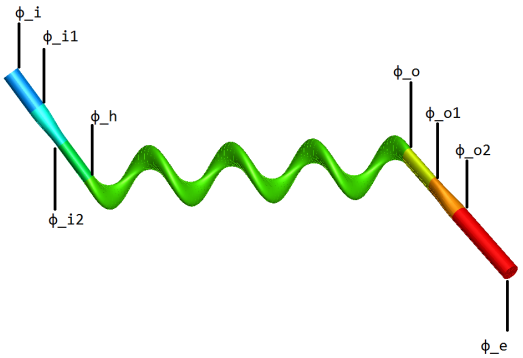


Figure 2.1: Model of a helical pipe. The dark blue is the circular section at the inlet, the light blue section is the transition from circle to ellipse, the light green section is where the pipe is straight with an elliptical cross-section, the green section is the helical section, the yellow section is the straight elliptical section near the outlet, the orange section is the transition from an ellipse to a circle and the red section is the straight circular outlet.

2.3 Pipe models

Constructing a helical shaped pipe in global Cartesian coordinates, is achieved using the `oomph-lib` mesh, `TubeMesh`. This mesh maps (ϕ, r, θ) to global Cartesian space. Here, ϕ is the centreline parameter, r is the radial parameter, defined such that $r = 0$ is the centreline and $r = 1$ is the boundary, and θ is the azimuthal angle, . This is achieved by first defining the pipe centreline in terms of ϕ and then creating a two-dimensional cross-section using r and θ. The global Cartesian representation of any point in the pipe is

$$\mathbf{x} = \mathbf{R}(\phi) - X(r,\theta)\mathbf{N}(\phi) + Y(r,\theta)\mathbf{B}(\phi) \tag{2.1}$$

where $\mathbf{x} = (x, y, z)$ is the global Cartesian coordinate, ϕ is the centreline parameter, $\mathbf{R}(\phi)$ is the centreline, X and Y are the functions that define the cross-section, $\mathbf{N}$ is the inwards pointing unit normal, and $\mathbf{B}$ is the unit bi-normal. The normal direction, $\mathbf{N}(\phi)$ and the binormal direction, $\mathbf{B}(\phi)$ are found using the Frenet forumlae,

$$\mathbf{T} = \frac{\mathrm{d}\mathbf{R}}{\mathrm{d}\phi}\frac{\mathrm{d}\phi}{\mathrm{d}s}, \quad \mathbf{N} = \frac{1}{\kappa}\frac{\mathrm{d}\mathbf{T}}{\mathrm{d}\phi}\frac{\mathrm{d}\phi}{\mathrm{d}s}, \quad \mathbf{B} = \mathbf{T} \times \mathbf{N}, \tag{2.2}$$

where $s(\phi)$ is the arc-length of the helix.

The centreline used for this research is a helix with straight sections at the inlet and outlet. In order to define the required centreline, a helix of

arbitrary origin is defined,

$$\mathbf{R}(\phi) = (R_c\cos(\phi-\phi_\mathrm{h}) + x_0, R_c\sin(\phi-\phi_\mathrm{h}) + y_0, \frac{H(\phi-\phi_\mathrm{h})}{2\pi} + z_0) \quad (2.3)$$

where R_c is the non-dimensional radius of the helix and H is the pitch of the pipe, as is shown in figure 2.3, and ϕ_h is the value of the centreline parameter at which the helix begins. The piecewise centreline for the geometry is determined by the origin of the centreline $(R_c, 0, 0)$, and by setting $\mathbf{R}(\phi)$ and $\frac{\mathrm{d}\mathbf{R}}{\mathrm{d}\phi}$ to be continuous everywhere. This gives a centreline of

$$\mathbf{R}(\phi) = \begin{cases} \mathbf{R}_\mathrm{i}(\phi), & 0 \le \phi < \phi_\mathrm{h}, \\ \mathbf{R}_\mathrm{h}(\phi), & \phi_\mathrm{h} \le \phi < \phi_\mathrm{o}, \\ \mathbf{R}_\mathrm{o}(\phi), & \phi_\mathrm{o} \le \phi \le \phi_\mathrm{e}, \end{cases} \quad (2.4)$$

where

$$\mathbf{R}_\mathrm{i}(\phi) = R_c\mathbf{i} + R_c\phi\mathbf{j} + \frac{H\phi}{2\pi}\mathbf{k} \quad (2.5)$$

is the centreline of the inlet,

$$\mathbf{R}_\mathrm{h}(\phi) = R_c\cos(\phi-\phi_\mathrm{h})\mathbf{i} + [R_c\sin(\phi-\phi_\mathrm{h}) + R_c\phi_\mathrm{h}]\mathbf{j} + \frac{H\phi}{2\pi}\mathbf{k} \quad (2.6)$$

is the centreline of the helical section, and

$$\begin{aligned}\mathbf{R}_\mathrm{o}(\phi) = {} & [R_c\cos(\phi_\mathrm{o}-\phi_\mathrm{h}) - R_c\sin(\phi_\mathrm{o}-\phi_\mathrm{h})(\phi-\phi_\mathrm{o})]\mathbf{i} \\ & + [R_c\sin(\phi_\mathrm{o}-\phi_\mathrm{h}) + R_c\phi_\mathrm{h} + R_c\cos(\phi_\mathrm{o}-\phi_\mathrm{h})(\phi-\phi_\mathrm{o})]\mathbf{j} + \frac{H\phi}{2\pi}\mathbf{k} \quad (2.7)\end{aligned}$$

is the centreline of the outlet. The value of the centreline at the beginning of the outlet is ϕ_o and the value of the centreline at the end of the outlet is ϕ_e. The geometry of the centreline of the helix is characterised by two non-dimensional parameters

$$\tau = \frac{\frac{H}{2\pi}}{R_c^2 + \left(\frac{H}{2\pi}\right)^2} \quad \text{and} \quad \kappa = \frac{R_c}{R_c^2 + \left(\frac{H}{2\pi}\right)^2} \quad (2.8)$$

and the arc-length of the helix is $s = \phi\sqrt{R_c^2 + (H/(2\pi))^2}$.

The cross-section of the helical pipe is determined by using r and θ to define a cross-section that can transition from a circle to an ellipse. The

function

$$F(\phi) = \begin{cases} 0 & \phi \leq \phi_{\mathrm{i}1}, \\ 6\left(\frac{\phi-\phi_{\mathrm{i}1}}{\phi_{\mathrm{i}2}-\phi_{\mathrm{i}1}}\right)^5 - 15\left(\frac{\phi-\phi_{\mathrm{i}1}}{\phi_{\mathrm{i}2}-\phi_{\mathrm{i}1}}\right)^4 + 10\left(\frac{\phi-\phi_{\mathrm{i}1}}{\phi_{\mathrm{i}2}-\phi_{\mathrm{i}1}}\right)^3 & \phi_{\mathrm{i}1} \leq \phi < \phi_{\mathrm{i}2}, \\ 1 & \phi_{\mathrm{i}2} \leq \phi \leq \phi_{\mathrm{o}1}, \\ 1 - \left(6\left(\frac{\phi-\phi_{\mathrm{o}1}}{\phi_{\mathrm{o}2}-\phi_{\mathrm{o}1}}\right)^5 - 15\left(\frac{\phi-\phi_{\mathrm{o}1}}{\phi_{\mathrm{o}2}-\phi_{\mathrm{o}1}}\right)^4 + 10\left(\frac{\phi-\phi_{\mathrm{o}1}}{\phi_{\mathrm{o}2}-\phi_{\mathrm{o}1}}\right)^3\right) & \phi_{\mathrm{o}1} \leq \phi < \phi_{\mathrm{o}2}, \\ 0 & \phi_{\mathrm{o}2} \leq \phi_{\mathrm{e}}, \end{cases} \tag{2.9}$$

is used to define the shape of the cross-section, where $\phi_{\mathrm{i}1}$ is the beginning of the transition from a circle to an ellipse, $\phi_{\mathrm{i}2}$ is the end of the transition to an ellipse, $\phi_{\mathrm{o}1}$ is the beginning of the transition from an ellipse to a circle, $\phi_{\mathrm{o}2}$ is the end of the transition to a circle and ϕ_{e} is the end of the pipe. This function is used as it smoothly changes from 0 to 1, as shown in figure 2.2. The functions $X(r,\theta)$ and $Y(r,\theta)$ are

$$X(r,\theta) = \{aF(\phi) + \sqrt{ab}[1 - F(\phi)]\}r\cos(\theta),$$

$$Y(r,\theta) = \{bF(\phi) + \sqrt{ab}[1 - F(\phi)]\}r\sin(\theta),$$

where a is the length of the normal axis of the ellipse and b is the length of the binormal axis of the ellipse. A pipe model in Cartesian space and the cross-sectional coordinates of the pipe are shown in figure 2.3. The parameter $e = a/b$ is used to define the aspect ratio of the cross-section. For $e = 1$, the cross-section is circular, for $e > 1$, the binormal axis of the ellipse is larger than the normal axis, and for $e > 1$, the normal axis of the ellipse is larger than the binormal axis.

The local coordinates X and Y are recovered from the global Cartesian coordinates by the transform

$$X = [\mathbf{x} - \mathbf{R}(\phi)] \cdot \mathbf{N}(\phi), \tag{2.10}$$

$$Y = [\mathbf{x} - \mathbf{R}(\phi)] \cdot \mathbf{B}(\phi), \tag{2.11}$$

and the centreline distance $Z = \phi\sqrt{R_c^2 + (H/(2\pi))^2} = s$. The subscript conventions used for ϕ are also used for Z. Similarly, the local velocities (U, V, W), where U is the component of velocity in the normal direction, V is the component of velocity in the binormal direction and W is the axial velocity, are found from the global Cartesian velocities (u, v, w), with

$$U = (u, v, w) \cdot \mathbf{N}(\phi), \tag{2.12a}$$

$$V = (u, v, w) \cdot \mathbf{B}(\phi), \tag{2.12b}$$

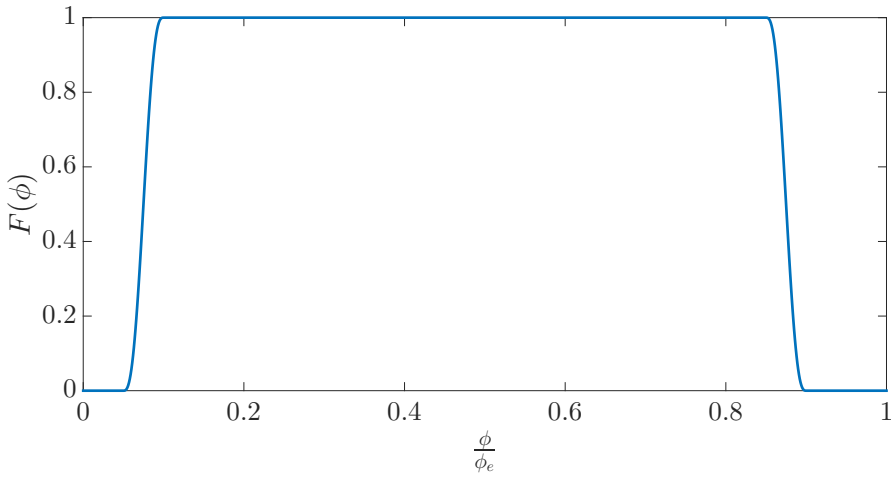


Figure 2.2: $F(\phi)$ over the length of the pipe.

$$W = (u, v, w) \cdot \mathbf{T}(\phi). \tag{2.12c}$$

Five helical centrelines are used for this research. The first two geometries replicate the experimental geometries reported by Levy et al. (2015). One of the experimental pipes has a larger dimensional cross-sectional diameter and is referred to as the large pipe, and the other pipe is referred to as the small pipe. The remaining three are representative of umbilical cords, with U_{t3} representing a normocoiled cord, U_{c2} representing a hypercoiled cord and U_{c7} representing a hypocoiled cord, and are taken from, and follow the naming convention of, Wilke (2016).

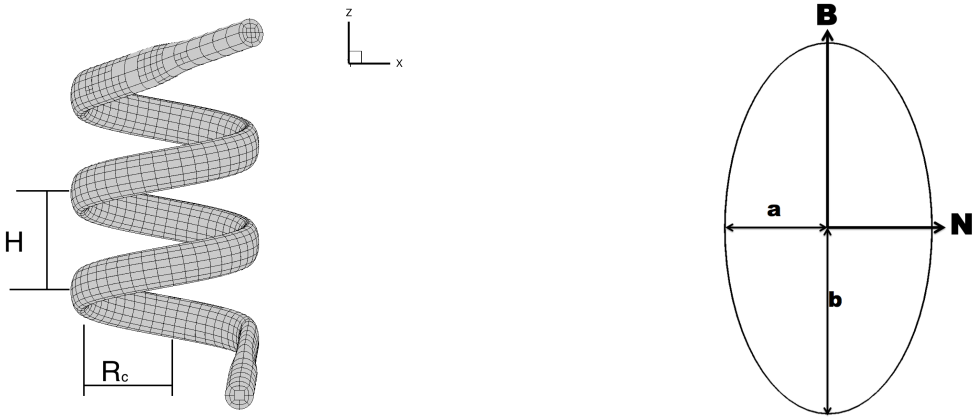
2.4 Finite-element method

This section is an introduction to the finite-element method (FEM), which is used to find an approximation to the Navier-Stokes equations for flow in helical geometries. An introduction to this method, as presented in Zienkiewicz et al. (2014), is shown below.

The finite-element method is a numerical method that attempts to find a weak solution to a differential equation. To introduce the finite-element method, the quasi-harmonic equation

$$-\frac{\partial}{\partial x_i}\left(k\frac{\partial u}{\partial x_i}\right) + Q = 0 \tag{2.13}$$

where k and Q are specified functions, is considered. This is equation is used to introduce the finite-element method as it is a very convenient example to



(a) Pipe model

(b) Cross-sectional coordinates

Figure 2.3: The model of the pipe in Cartesian coordinates and the cross-section of the pipe in **N**-**B** space.

introduce the numerical approximations. The boundary conditions can be either Dirichlet type,

$$u = \bar{u} \quad \text{on} \quad \Gamma_u \tag{2.14}$$

or Neumann type,

$$q_n = -k\frac{\partial u}{\partial n} = \bar{q}_n \quad \text{on} \quad \Gamma_q \tag{2.15}$$

where a bar denotes a specified quantity, Γ_u is the section of the boundary with Dirichlet boundary conditions and Γ_q is the section of the boundary with Neuman conditions. Equations (2.13) to (2.15) are defined as the *strong form* of the problem, and u is a strong solution.

Direct use of equation (2.13) requires computation of the second derivative to solve the problem. This requirement is weakened by considering an integral expression for equation (2.13) written as

$$\int_\Omega v\left[-\frac{\partial}{\partial x_i}\left(k\frac{\partial u}{\partial x_i}\right) + Q\right] \mathrm{d}\Omega = 0 \tag{2.16}$$

where v is an arbitrary function and Ω is the domain.

By integrating by parts the second derivative terms equation (2.16), we obtain

$$\int_\Omega \frac{\partial v}{\partial x_i}\left(k\frac{\partial u}{\partial x_i}\right) \mathrm{d}\Omega + \int_\Omega v\,Q\,\mathrm{d}\Omega - \int_\Gamma v n_i \left(k\frac{\partial u}{\partial x_i}\right) \mathrm{d}\Gamma = 0. \tag{2.17}$$

The boundary is split into two parts, Γ_u and Γ_q, with $\Gamma = \Gamma_u \cup \Gamma_q$ and $\Gamma_u \cap \Gamma_q = 0$. Using equation (2.15) in (2.17) gives

$$\int_\Omega \frac{\partial v}{\partial x_i}\left(k\frac{\partial u}{\partial x_i}\right)\mathrm{d}\Omega + \int_\Omega v\,Q\,\mathrm{d}\Omega + \int_{\Gamma_q} v\bar{q}_n\,\mathrm{d}\Gamma = 0, \tag{2.18}$$

which is valid only if v vanishes on Γ_u. Hence equation (2.14) must be imposed for equivalence. Equation (2.18) is known as the weak form of the problem since one less derivative order is necessary for constructing the solution. Weak forms are the basis for the finite-element solutions.

The Galerkin method approximates the dependent variable u by writing u as a sum of known *trial functions* $N_a(x_i)$ and unknown parameters $\tilde{u}^a$. Thus, it is written as

$$u \approx \hat{u} = \sum_{a=1}^{n} N_a(x_i)\tilde{u}^a. \tag{2.19}$$

In a similar way, the arbitrary variable v is expressed as

$$v \approx \hat{v} = \sum_{a=1}^{n} W_a(x_i)\tilde{v}^a, \tag{2.20}$$

where W_a are *test functions* and $\tilde{v}^a$ are arbitrary parameters. This method converts equation (2.18) to a set of algebraic equations.

Each of the test and trial functions are defined in partitions Ω_e of the total domain Ω. This division is denoted by

$$\Omega \approx \Omega_h = \bigcup \Omega_e, \tag{2.21}$$

where the Ω_e are known as elements. Elements are discussed in further detail in section 2.5. The test and trial functions are linear polynomials in each element and the unknown parameters are *nodal values* of u. A node is a point in the element, usually located at a corner or end-point.

The Galerkin method sets $W_a = N_a$, which gives the assembled system of equations

$$\sum_{a=1}^{n} K_{ba}\tilde{u}^a + f_b = 0, \quad b = 1, 2, ..., n - r \tag{2.22}$$

where b denotes a specific node, r is the number of nodes appearing in the approximation of the Dirichilet boundary condition and K_{ba} is assembled from element contributions K^e_{ba} with

$$K^e_{ba} = \int_{\Omega_e} \frac{\partial N_b}{\partial x_i} k \frac{\partial N_a}{\partial x_i}\mathrm{d}\Omega. \tag{2.23}$$

Similarly f_b is computed from the elements as

$$f_b^e = \int_{\Omega_e} N_b Q \mathrm{d}\Omega + \int_{\Gamma_{eq}} N_b \hat{q}_n \mathrm{d}\Gamma. \tag{2.24}$$

To impose Dirichlet boundary conditions, $\tilde{u}^b = \bar{u}^b$, and $\tilde{u}^a = \bar{u}^a$ are set for each boundary node.

2.5 oomph-lib

The finite-element method is implemented using `oomphlib`, an **o**bject-**o**riented **m**ulti-**ph**ysics **lib**rary, developed at the University of Manchester (Heil & Hazel 2017). `oomph-lib` is an open-source finite-element library written in `c++` and was originally designed to find solutions to physiological fluid dynamic problems. `oomph-lib` uses Newton's method to solve the non-linear system of equations given by the Galkerin method. In this section, the elements and mesh generated by `oomph-lib` will be discussed.

In the finite-element method, an element is a small partition of the domain of the problem. The points on an element where the function is evaluated are called nodes. In three dimensions, elements are often tetrahedral or hexahedral shapes with nodes at the vertices and possibly at the midpoints of the edges. For helical pipe models, `oomph-lib` uses hexahedral elements with a $3\times3\times3$ -node arrangement. A typical element can be seen in figure 2.4.

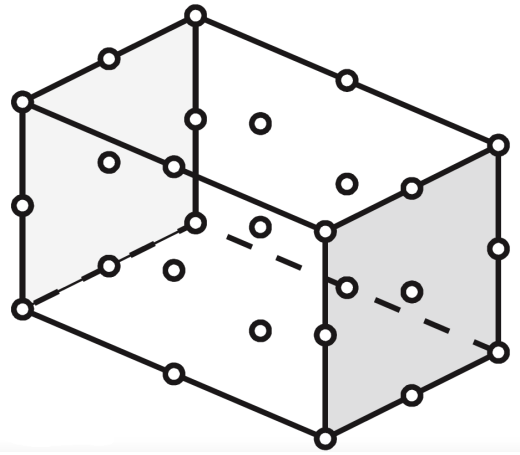


Figure 2.4: A hexahedral element with 27 nodes, taken from Zienkiewicz et al. (2013). Each small circle represents a node.

There are two elements that can be used to solve Navier-Stokes equations in `oomph-lib`. They are the Taylor-Hood and Crouzeix-Raviart elements.

The elements compute the same velocity and use the same shape functions. The three-dimensional shape functions, ψ, used to compute the velocity are found by multiplying three one-dimensional shape functions, Ψ. The one-dimensional shape functions used by the elements are

$$\Psi_1(s_i) = \frac{1}{2}s_i(s_i - 1), \tag{2.25}$$

$$\Psi_2(s_i) = 1 - s_i^2, \tag{2.26}$$

$$\Psi_3(s_i) = \frac{1}{2}s_i(s_i + 1), \tag{2.27}$$

where s_i are the local element coordinates. There are 27 possible combinations of the one-dimensional shape functions, with each combination corresponding to the three-dimensional shape function for a node. The three-dimensional shape functions are

$$\psi_{i,j,k} = \Psi_j(s_1)\Psi_k(s_2)\Psi_l(s_3) \tag{2.28}$$

for $j, k, l = 1, 2, 3$.

The elements differ in the calculation of the pressure field. Taylor-Hood elements interpolate tri-linearly over the values at the vertex nodes, and the pressure field is given by

$$p = \sum_{j=1}^{N_p^{(E)}} P_j^{(E)}\psi_j^{(p)} \tag{2.29}$$

where E is the element number, $P_j^{(E)}$ are the pressure values stored at the elements' $N_p^{(E)} = 8$ corner nodes and $\psi_j^{(p)}$ are the tri-linear pressure shape functions. Crouzeix-Raviart elements store four pressure degrees of freedom within each element. The pressure at any point within the element is given by

$$p = P_0^{(E)} + P_1^{(E)}s_1 + P_2^{(E)}s_2 + P_3^{(E)}s_3 \tag{2.30}$$

where $P_i^{(E)}$ is the i-th pressure degree of freedom in element E and s_i are the elements local coordinates. Although the Taylor-Hood elements do not conserve mass locally, they do ensure that the pressure field is continuous across elements. Discontinuities will be present across elements in Crouzeix-Raviart elements. Also, for sufficiently refined meshes Taylor-Hood elements generate a smaller number of pressure degrees of freedom. For these reasons, Taylor-Hood elements are used in this research.

2.5.1 Error tolerance

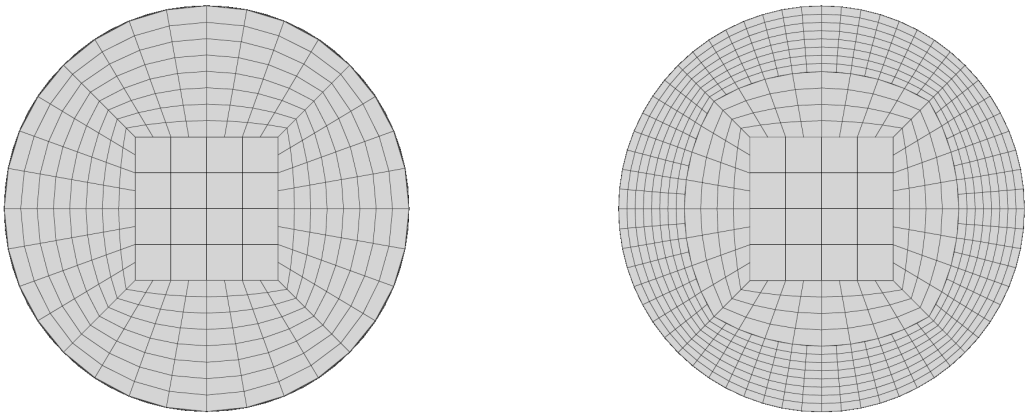
To evaluate the error of the finite-element solution, oomph-lib uses a Z2-error-estimator. The estimator uses the fact that the finite-element representation of the flux,

$$f_i^{[FE]} = \sum_{j=1}^{N} U_j \frac{\partial \psi_j}{\partial x_i}, \tag{2.31}$$

where U_j is the solution at node j and ψ_j are the global basis functions of the solution, is discontinuous between elements. The magnitude of the discontinuity decreases under mesh refinement. Z2-error estimation computes a continuous and more accurate value of the flux, $f_i^{[rec]}$, by projecting the finite-element based flux $f_i^{[FE]}$ onto a set of continuous basis functions known as recovery shape functions. The normalised elemental error is then estimated by

$$E_e = \sqrt{\frac{\int_e (f_i^{[rec]} - f_i^{[FE]})^2 \mathrm{d}v}{\sum_{e'}^{N_e} \int_{e'} (f_i^{[rec]})^2 \mathrm{d}v}} \quad \text{for} \quad e = 1, ..., N_e, \tag{2.32}$$

where N_e is the number of elements. Regions of high error are adapted to include more elements to reduce error. Regions of low error are adapted to have less elements to decrease computation time. The user defines a tolerance that defines the limits of high and low error. An adapted mesh is presented in figure 2.5. This mesh is used in the unsteady verification in section 2.7 and is used to solve a flow with a sharp velocity gradient near the wall. The highest errors are found near the pipe wall, and more elements are needed to resolve the flow, so the elements near the wall are adaptively refined.



(a) Loose tolerance (b) Tight tolerance

Figure 2.5: Comparison of mesh used in section 2.7 at loose and tight tolerance for $Re = 400$ and $t/T = 0$ showing adaptive refinement near the pipe walls at the tight tolerance.

2.6 Steady verification

Germano (1989) finds a solution for steady, fully developed flow in an infinitely long helical pipe. The velocity field is found by considering a perturbation of Poiseuille flow in small non-dimensional parameter $\epsilon = \hat{\kappa}\hat{b}$, and taking an expansion in integer powers of the Dean number $De = 2\epsilon Re^2$. Using this method, Germano gives the Cartesian components of secondary motion, U and V, explicitly. These are given in appendix A. As this solution is approximate, the numerical solution and asymptotic solution are compared at small ϵ and De. The numerical and asymptotic solutions are compared by calculating the cross-flow energy

$$E = U^2 + V^2 \tag{2.33}$$

from the cross-flow velocity components in a cross-sectional plane. The cross-sectional plane chosen to compare these results is sufficiently far downstream such that the flow is fully developed, but far away enough from the outlet for there to be no noticeable downstream boundary effects.

2.6.1 The computational model

The domain consists of a straight inlet and outlet of length $5D$ and a helical coil with parameters given in table 2.1. These parameters are chosen so that

both $e < 1$ and $e > 1$ are tested, and so the helical centreline creates a pipe where the solution presented in Germano (1989) is applicable. The pipe used in the $e < 1$ case is shown in figure 2.6.

R_c	H	κ	τ	Z_e	e	Re	ϵ	De
5	18.14	0.150	0.0866	50	0.6	40	0.0968	310
5	18.14	0.150	0.0866	50	1.4	40	0.0634	203

Table 2.1: Dynamic and geometric parameters for pipes used for verification.

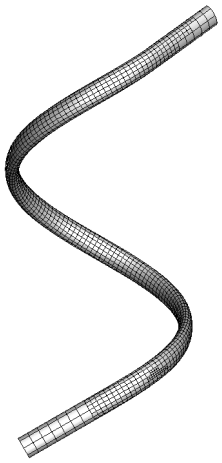


Figure 2.6: One of the pipes used for steady-flow verification. The geometric parameters are $R_c = 5$, $H = 18.14$, $\kappa = 0.150$, $\tau = 0.0866$, $Z_e = 50$ and $e = 0.6$.

The inlet boundary condition is the exact solution for flow through a straight pipe with an elliptical cross-section, shown in equation (1.10). The no-slip, $U = V = W = 0$, condition is used on the wall. The outlet is taken to be pseudo traction-free. A pseudo traction-free condition weakly enforces a zero axial derivative. This is favourable for pipes with straight ends as the flow naturally becomes fully developed.

2.6.2 Results

In order to compare the finite-element solution to the perturbation solution presented in Germano (1989), the cross-flow energy, E, is calculated for three different error tolerances, shown in table 2.2, in `oomph-lib` and compared

graphically to the perturbation solution. The solution given by (Germano 1989) is not exact, so a graphical comparison is chosen over a pointwise comparison. The cross-sectionally averaged cross-sectional energy for the tightest error tolerance is then compared at a range of Dean numbers with the perturbation solution and is also shown along the axial distance for the tightest error tolerance and highest Reynolds number.

tolerance	minimum	maximum
loose	10^{-3}	10^{-4}
medium	5×10^{-4}	5×10^{-5}
tight	2×10^{-4}	2×10^{-5}

Table 2.2: Table of error tolerances.

For sufficiently low values of ϵ and Re, tightening the error tolerance of the solver results in the finite-element method solution approaching the asymptotic solution from Germano (1989). Graphical inspection of figures 2.7 and 2.8 shows that as the error tolerance of the solver becomes tighter, the solution converges on the asymptotic solutions shown in figure 2.7d and 2.8d. A visual comparison of figures 2.7c and 2.7d, and 2.8c and 2.8d show that the 'tight' tolerance is sufficient for both $e = 0.6$ and $e = 1.4$.

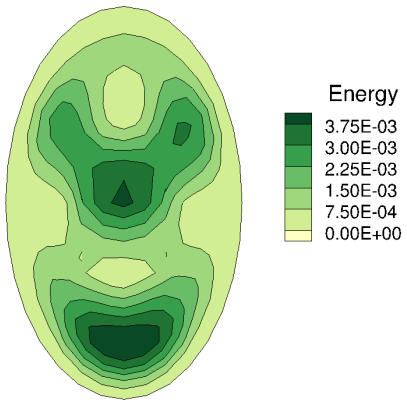
Both pipes are long enough for fully-developed flows to form. Figure 2.9 shows the cross-sectionally averaged cross-flow energy

$$\bar{E} = \frac{1}{A} \iint_A E(A') \mathrm{d}A' \tag{2.34}$$

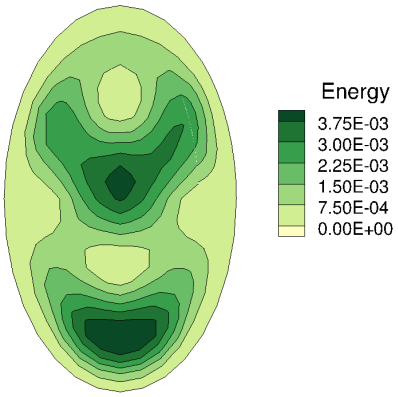
as a function of centreline arclength for two pipes that have the same centreline geometry but different values of e. The centreline parameters are shown in table 2.1. The flow is fully developed from Z = 18 to Z = 44, where the average cross-flow energy changes by less than 2%. The aspect ratio does not change the development length of the cross-flow energy. The peak average energy occurs at the transition from the straight pipe to a helical pipe, where there is a sudden change in curvature. The energy then settles to the fully developed magnitude. For lower Reynolds numbers, the flow becomes fully developed earlier in the pipe. This is consistent with results for straight pipes, as the development length is smaller for flows with smaller Reynolds numbers.

The cross-sectionally averaged cross-flow energy is calculated from both the solution presented in Germano (1989) and from the finite-element results. The cross-sectionally averaged energy for both solutions is compared for six values of De corresponding to $Re = 20, 40, 60, 80, 100$ and 120. For low Dean

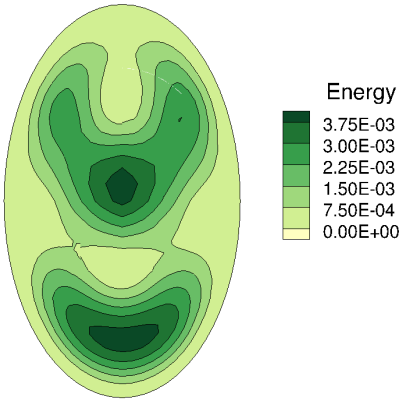
number, the cross-sectionally averaged cross-flow energy is very similar for both the finite-element solution and the asymptotic solution. This can be seen in figures 2.10 and 2.11. The computed solution and analytical solution show very good agreement when the analytical solution is most valid. This is when both ϵ and De are low. As the Reynolds number increases, the average energy of the finite-element solution increases at a decreasing rate. It is noteworthy that Germano's solution remains valid for De as high as about 500 for small values of ϵ.



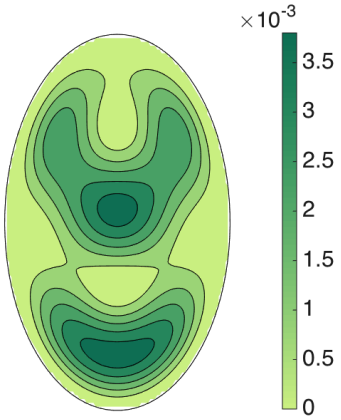
(a) Loose tolerance



(b) Medium tolerance

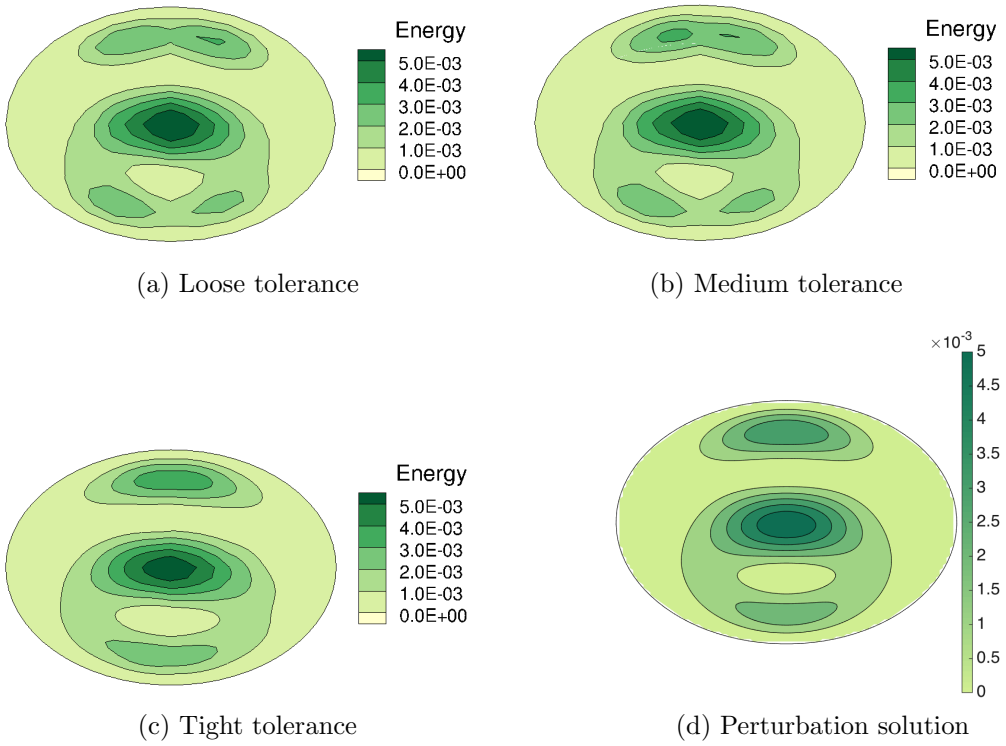


(c) Tight tolerance



(d) Perturbation solution

Figure 2.7: Cross-flow energy at $L = 35$ for $e = 0.6$, $\epsilon = 0.0968$, $De = 310$. (a) - (c) show the computed solution and (d) shows the solution from Germano (1989).



(a) Loose tolerance

(b) Medium tolerance

(c) Tight tolerance

(d) Perturbation solution

Figure 2.8: Cross-sectional energy at $L = 35$ for $e = 1.4$, $\epsilon = 0.0634$, $De = 203$. (a) - (c) show the computed solution and (d) shows the solution from Germano (1989).

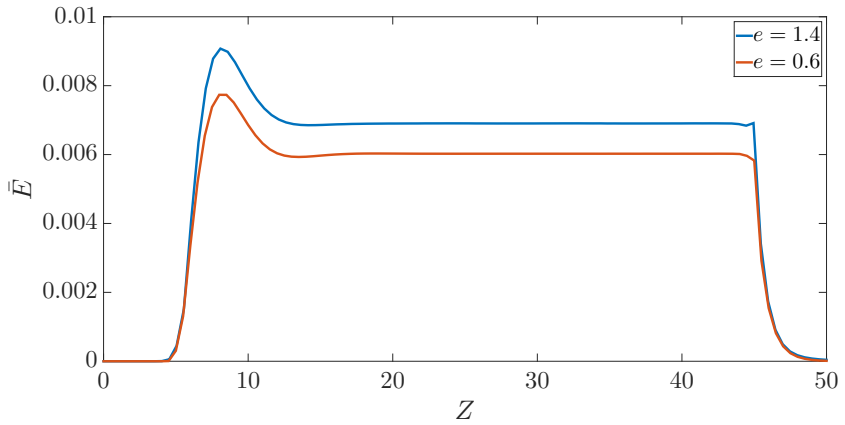


Figure 2.9: Cross-sectionally averaged cross-sectional energy along length of pipe defined in table 2.1 for $Re = 120$. The helical section starts at $Z = 5$ and ends at $Z = 45$.

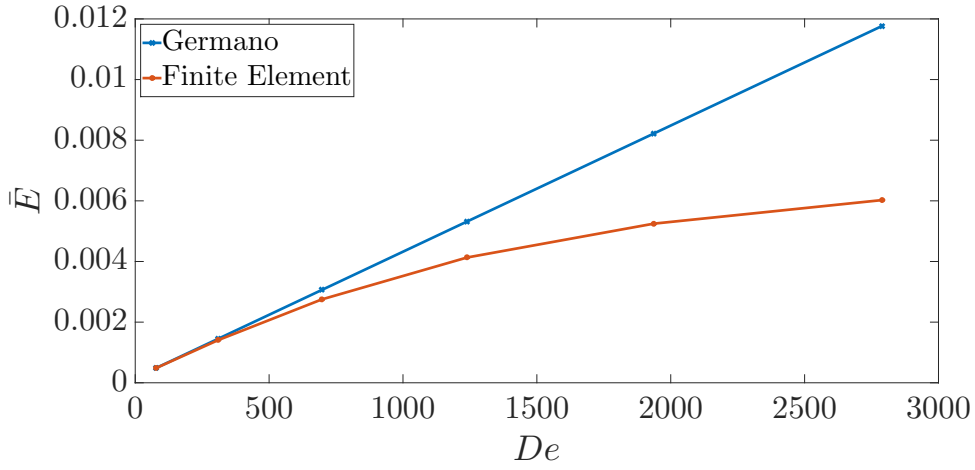


Figure 2.10: Cross-sectionally averaged cross-flow energy for the pipe defined in table 2.1 for $e = 0.6$ at $Z = 35$.

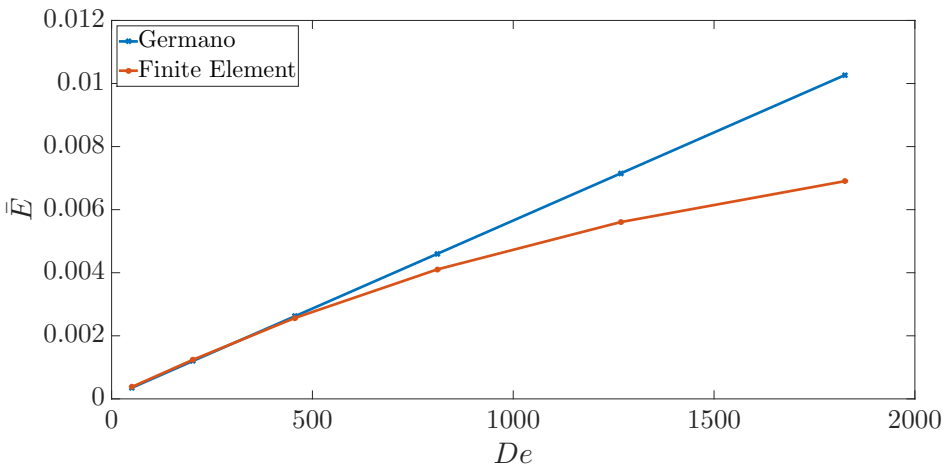


Figure 2.11: Cross-sectionally averaged cross-flow energy for the pipe defined in table 2.1 for $e = 1.4$ at $Z = 35$.

2.7 Unsteady verification

2.7.1 Background

To verify the numerical method used to compute unsteady flows, pulsatile Womersley flow through a straight pipe is considered. The straight pipe is oriented such that the axial direction of the pipe is in the z direction. The components of velocity in the x and y directions are both zero. The exact solution for an infinitely long straight pipe derived by Womersley (1955) is used as an boundary and initial condition. The temporal variation of the cross-sectionally averaged axial velocity is defined as the addition of a mean and two harmonic terms,

$$\bar{w}(t) = \langle \bar{w} \rangle (1 + a_1 \sin(\omega_1 t) + a_2 \cos(\omega_2 t)) \tag{2.35}$$

where $a_1, a_2 \in \mathbb{R}$ and $\langle \bar{w} \rangle$ is the temporally and cross-sectionally averaged axial velocity within the cross-section. This is the same as that used in Sherwin & Blackburn (2005) and is used here as it is has been previously used to model physiological flow. The choices of a_1, a_2, and $\langle \bar{w} \rangle$ are used to prescribe the velocity variation and determine the requisite P_n.

The cross-sectionally averaged axial velocity is also be found by averaging the fully velocity profile from equation (1.4),

$$\bar{w}(t) = \frac{2}{R^2} \int_0^R \left(W_0(r) + \sum_{n=1}^{N} \mathrm{Re} \left[\frac{P_n i}{\rho \omega_n} \left(\frac{J_0\left(i^{\frac{3}{2}} \alpha_n \frac{r}{R}\right)}{J_0\left(i^{\frac{3}{2}} \alpha_n\right)} - 1 \right) e^{i\omega_n t} \right] \right) r \mathrm{d}r, \tag{2.36}$$

where $W_0(r) = 2[1 - (r/R)^2]$ and R is the pipe radius. Using the integral identity,

$$\int_0^\beta \theta J_0(\theta) \mathrm{d}\theta = \beta J_1(\beta), \tag{2.37}$$

and taking the real part of the integral following the work of Mazumdar (1992), equation (2.36) can be simplified. After integration, equation (2.36) becomes

$$\bar{w}(t) = w_{\mathrm{mean}} + \sum_{n=1}^{N} \mathrm{Re} \left(\frac{P_n i}{\rho \omega_n} \left(\frac{2 J_1(\beta_n)}{\beta_n J_0(\beta_n)} - 1 \right) e^{i\omega_n t} \right) \tag{2.38}$$

where $\beta_n = i^{\frac{3}{2}} \alpha_n$ and w_{mean} is the mean of the steady component of the flow. The mean component of the steady flow is equated to the cross-sectionally and time-averaged velocity $\langle \bar{w} \rangle$.

Writing the complex pressure coefficients $P_n = C_n + iD_n$ with $n = 1, 2$, and equating equations (2.35) and (2.38) gives

$$C_1 = -\frac{a_1 \rho \omega_1 W_{\text{mean}} \operatorname{Re}\left(\frac{2J_1(\beta_1)}{\beta_1 J_0(\beta_1)} - 1\right)}{\left|\left|\frac{2J_1(\beta_1)}{\beta_1 J_0(\beta_1)} - 1\right|\right|^2}, \tag{2.39}$$

$$D_1 = \frac{a_1 \rho \omega_1 W_{\text{mean}} \operatorname{Im}\left(\frac{2J_1(\beta_1)}{\beta_1 J_0(\beta_1)} - 1\right)}{\left|\left|\frac{2J_1(\beta_1)}{\beta_1 J_0(\beta_1)} - 1\right|\right|^2}, \tag{2.40}$$

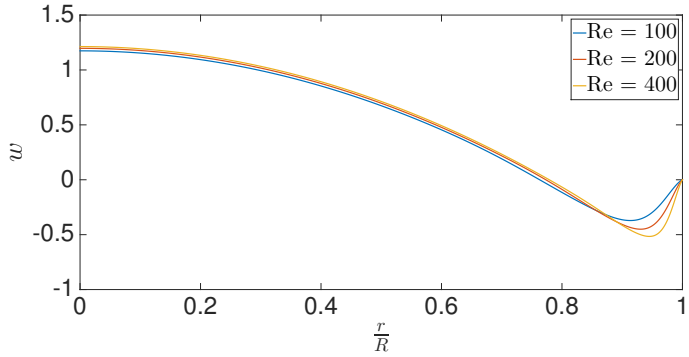
$$C_2 = -\frac{a_2 \rho \omega_2 W_{\text{mean}} \operatorname{Im}\left(\frac{2J_1(\beta_2)}{\beta_2 J_0(\beta_2)} - 1\right)}{\left|\left|\frac{2J_1(\beta_2)}{\beta_2 J_0(\beta_2)} - 1\right|\right|^2}, \tag{2.41}$$

and

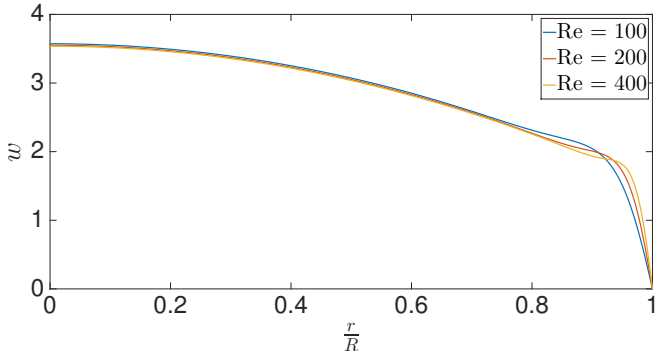
$$D_2 = -\frac{a_2 \rho \omega_2 W_{\text{mean}} \operatorname{Re}\left(\frac{2J_1(\beta_2)}{\beta_2 J_0(\beta_2)} - 1\right)}{\left|\left|\frac{2J_1(\beta_2)}{\beta_2 J_0(\beta_2)} - 1\right|\right|^2}. \tag{2.42}$$

2.7.2 Computational model

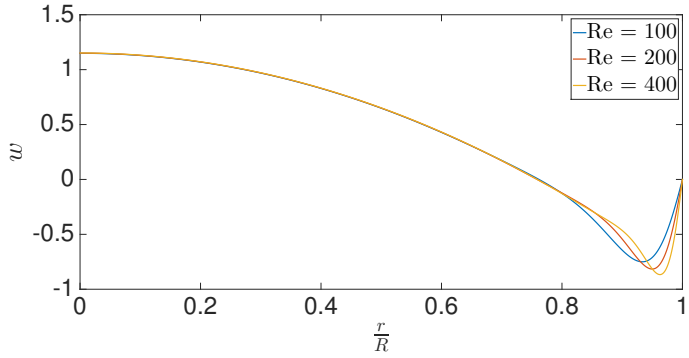
To verify the unsteady solver, pulsatile flow through straight pipes is calculated and compared to the exact solution at each node. This is done for $Re = 100, 200$ and 400 and $St = 1$. These Reynolds numbers are chosen because the required Reynolds numbers for investigating umbilical cords are around $Re = 200$. Increasing the Reynolds number increases the gradient of the flow profile near the walls. A Strouhal number of $St = 1$ is significantly larger than what will be required for simulating flows in umbilical cords, and will therefore lead to larger temporal gradients. The solver is verified at larger temporal gradients than necessary to ensure that the time step chosen is adequate for the convergence of the Newtonian solver. The time step used for verification is $\Delta t/T = 1/40$. For verification purposes, $a_1 = 0.75$ and $a_2 = -0.75$ are chosen, as it is a realistic approximation of physiologic waveforms (Sherwin & Blackburn 2005). Figure 2.12 shows the velocity profile for each Reynolds number chosen for verification at four time steps in the period. Figure 2.13 shows the cross-sectionally averaged axial flow over one period.



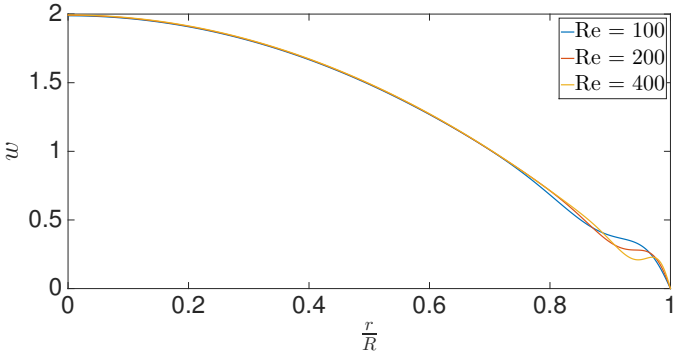
(a) $\frac{t}{T} = 0$



(b) $\frac{t}{T} = 0.25$



(c) $\frac{t}{T} = 0.54$



(d) $\frac{t}{T} = 0.75$

Figure 2.12: Inlet profile for unsteady flows for $Re = 100, 200$ and 400 at $t/T = 0$, 0.25, 0.54 and 0.75.

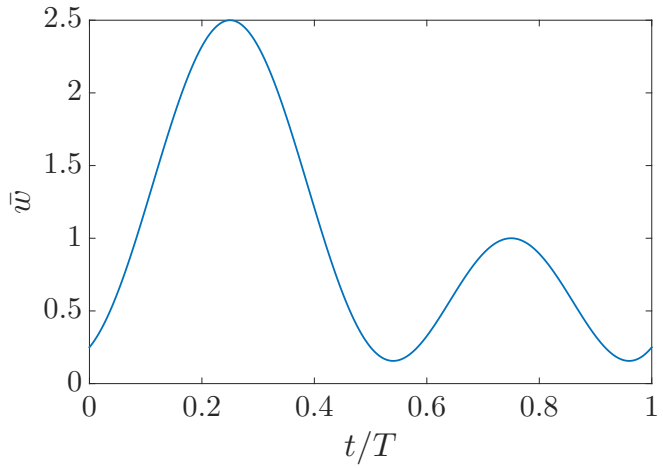


Figure 2.13: Cross-sectional averaged velocity over one period of flow.

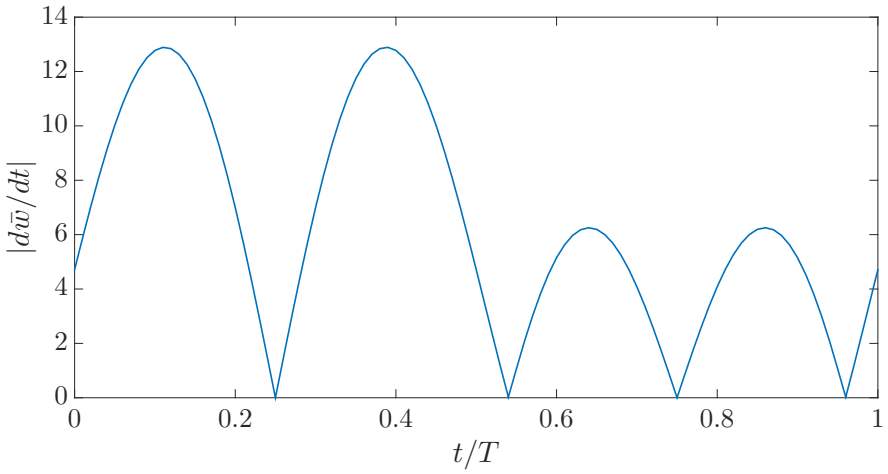


Figure 2.14: Absolute rate of change of cross-sectional averaged velocity over one period of flow.

The initial condition is set to be the exact solution at $t = 0$ and the boundary at the inlet is the exact solution at the required t. Similar to the steady flow case, there is a no slip condition on the pipe wall and the outlet is pseudo traction-free.

2.7.3 Results

The exact solution is compared with the finite-element solution and the error is considered over two periods of the flow. The solver is started without a time

Tolerance	Minimum	Maximum
Loose	4×10^{-3}	4×10^{-4}
Medium	2×10^{-3}	2×10^{-4}
Tight	10^{-3}	10^{-4}

Table 2.3: Table of error tolerances

history, so there is a numerical transient that settles down over time. The error is calculated by calculating the absolute value of the difference between the exact solution, $\sqrt{(w_{\text{FEM}} - w_{\text{exact}})^2}$, at every point that the computed solution is calculated. Figure 2.15 shows the absolute error for this flow along a slice taken at the mid-plane $y = 0$. The maximum error is found near the pipe walls where the velocity gradient is sharp. Through most of the slice, the error is $< 2.5\%$ of the bulk velocity. The volume-averaged error

$$\overline{\overline{\text{Error}}} = \iiint_V \text{Error}(x, y, z)\, \mathrm{d}V \tag{2.43}$$

is calculated over two periods of flow for each Reynolds number and tolerance.

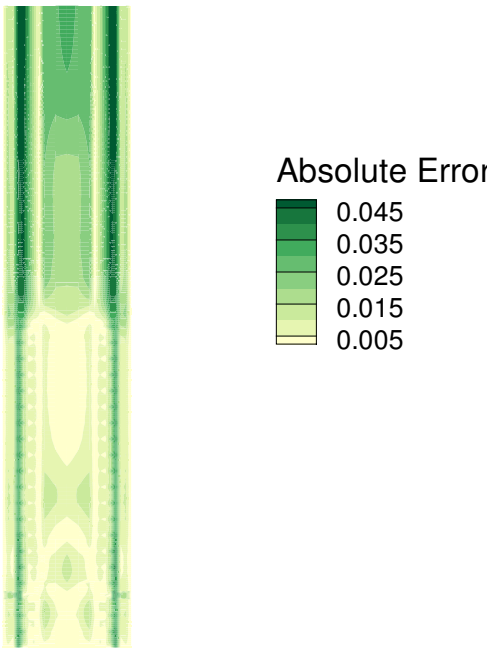


Figure 2.15: Absolute error at the mid-plane $y = 0$ for $Re = 100$, $St = 1$, $t = 20$, $t/T = 0$ and medium error tolerance.

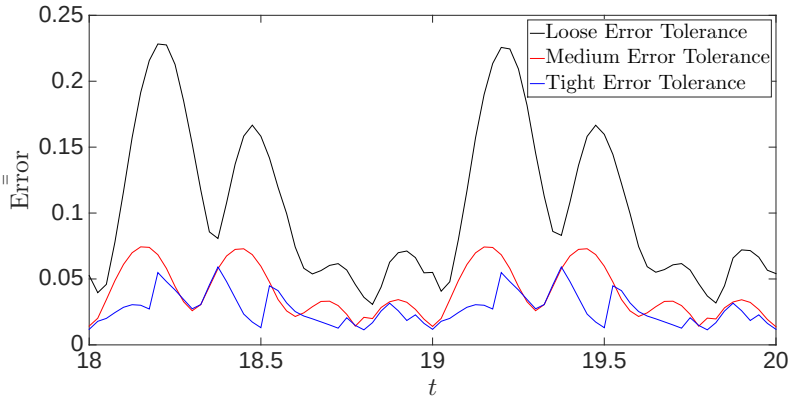


Figure 2.16: Volume-averaged error for $Re = 100$, $St = 1$.

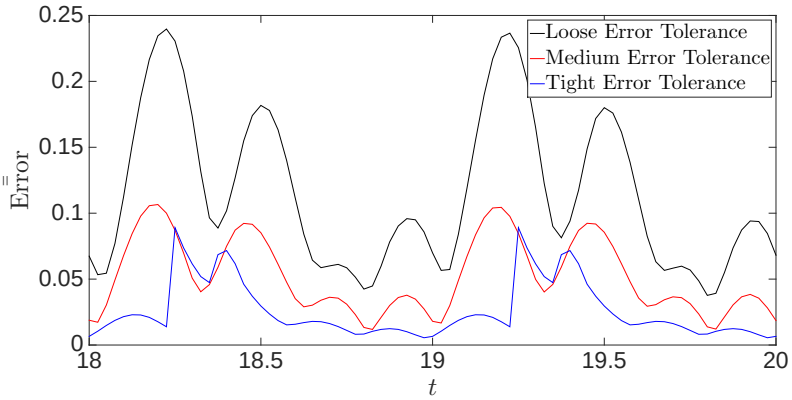


Figure 2.17: Volume-averaged error for $Re = 200$, $St = 1$.

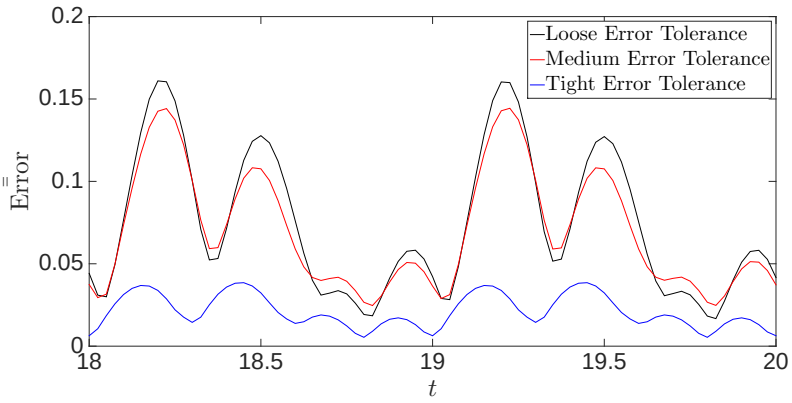


Figure 2.18: Volume-averaged error for $Re = 400$, $St = 1$.

tolerance	$Re = 100$	$Re = 200$	$Re = 400$
loose	0.105	0.115	0.0743
medium	0.0407	0.0526	0.0703
tight	0.0276	0.0266	0.0214

Table 2.4: Time and volume-averaged errors for unsteady verification runs.

Figures 2.16, 2.17 and 2.18 show the volume-averaged error for $Re = 100$, $Re = 200$ and $Re = 400$ respectively. The error tolerances used in the finite-element solver are shown in table 2.3. The general trend of the errors loosely follows the trend of $|d\bar{w}/dt|$, shown in figure 2.14. The error is highest in the first half of the period, and has two peaks, similar to $|d\bar{w}/dt|$. The error in the second half of the period is smaller, but also shows two smaller peaks. Errors for loose tolerance range from 3% to 23%, medium tolerance errors vary from 1% to 14% and the errors from the tight runs range from 1% to 8%. Table 2.4 shows the errors decreasing as the tolerance becomes tighter for each Reynolds number. This indicates that the solution is converging to the discrete-time problem.

For parameters typical of those investigated in this thesis, this verification study shows that a tight tolerance of Minimum $= 10^{-3}$ and Maximum $= 10^{-4}$ is needed to ensure sufficient accuracy. However, unsteady flow through a straight pipe is much simpler than flow through a helical pipe with elliptical cross-section because there is no secondary flow. In chapters 3 and 4 it is found that even tighter tolerances are sometimes needed to obtain adequately converged results.

2.8 Summary

When the error tolerance for a simulation does not lead to a spatially resolved solution, there are graphical indications that can be identified by inspecting the axial flow and cross-flow. Figure 2.8 shows that, at the medium tolerance, there are two lobes of higher E at the bottom of the pipe, and these merge in to one lobe at the higher tolerance. This indicates that the medium tolerance is insufficient, as tightening the tolerance of the solver visibly changes the solution. Similarly, at insufficient tolerances, the contours of the axial flow are jagged and not smooth. The contours of the flow become smoother with higher tolerances. Once smooth, the contours do not visibly change when the tolerance is tightened further, indicating that the solution has converged to graphical accuracy. Another indicator that the flow is not spatially resolved is that there are very small variations in both the axial and cross-flow along the

length of the pipe. This variation appears as a very low magnitude oscillation on a plot of $\overline{E}$ against Z. Tightening the tolerance further eliminates these small variations, and when there a no visible variations, the solution has converged to graphical accuracy. The results shown in Chapters 3 and 4 do not have these features, and are therefore considered to be spatially resolved.

Chapter 3

Comparison with experiments

Flows through helical pipes have been studied experimentally in the work of Levy et al. (2015). In this experiment, two straight plastic pipes were coiled in to the shape of a helix and a peristaltic pump was used to create very controlled flow rates in the pipe. Particle Image Velocimetry techniques were used to measure and calculate the cross-flow and vorticity of the flow. The experiment investigated steady flows as well as the effect of suddenly stopping steady flow. Of the two pipes used in the experiment, the cross-section of one was deformed in the coiling process, creating a cross-section that is roughly elliptical.

The aim of this chapter is to compare the experimental results with simulated results. This is achieved by simulating the two pipes in `oomph-lib`. The shapes of the velocity and vorticity fields of both the simulated and experimental results compare favourably in both the steady and suddenly stopping cases. The magnitudes of velocity and vorticity differ by an order of magnitude, despite both the experimental and computational results both using the average flow velocity to non-dimensionalise the velocity and vorticity (Levy et al. 2015). This discrepancy is discussed further in chapters 3.2 and 3.3, however a resolution to this discrepancy could not be found.

3.1 Pipe geometry

The parameters of the experimental pipes are shown in table 3.1, and the simulated models of each pipe are shown in figure 3.1. Both pipes are of moderate curvature and low torsion. The geometries used for simulations have the same non-dimensional parameters as the experimental pipes but have a smaller centreline length. The lengths chosen for the simulated pipes are shown in table 3.2 and are sufficient for the flow to become fully devel-

Pipe	R_c	H	κ	τ	e
Small coil	4.19	2.44	0.24	0.022	1
Large coil	2.96	3.43	0.33	0.061	0.5

Table 3.1: Experimental and simulated pipe parameters.

Z_i	Z_i1	Z_i2	Z_h	Z_o	Z_o1	Z_o2	Z_e
0	3.5	7	10.5	59.5	63	66.5	70

Table 3.2: Important pipe lengths of simulated pipes.

oped in the helical section. The experimental results contain time-averaged results for flow with a steady inlet. To best approximate this, simulations are performed with the steady solver, using the boundary conditions in table 3.3. Table 3.4 shows the tolerances and number of elements used to generate the results shown below, to indicate the computational cost of the simulations.

For the convenience of the reader, the origin of the pipe is moved to the centre of the helix, so that the slice in line with the diameter is now at $y = 0$. The reader is reminded that (x, y, z) are the global Cartesian coordinates and (X, Y, Z) are the local normal, binormal and centreline coordinates. Similarly (u, v, w) are the global Cartesian velocities and (U, V, W) are the local normal, binormal and centreline velocities. The experimental results presented in Levy et al. (2015) are measured in a plane that contains the central helical axis (Denier, personal communication, 2017). This does not coincide with the cross-section of the pipe orthogonal to the centreline. Any plane that contains the central helical axis will have a component of the axial flow in both the flow perpendicular to the plane, and the flow within the plane. Because the cross-flow is 1.3 to 5 times smaller than the axial flow, this misalignment noticeably changes the results. Figure 3.2 compares the flow in a cross-section perpendicular to the pipe centreline with the flow within a plane that contains the central helical axis. The vector field that results by considering the x-z plane has a component of axial flow distorting the field. However, the vector field in the middle of the experimental pipe in figure 3.5f is more horizontal than the simulated u-w vector field shown in figure 3.2a. The experimental vector field has features from both the u-w vector field, and the U-V vector field. The plane that the experimental results are taken from is likely between a plane that contains the central helical axis, and a plane perpendicular to the pipe centreline.

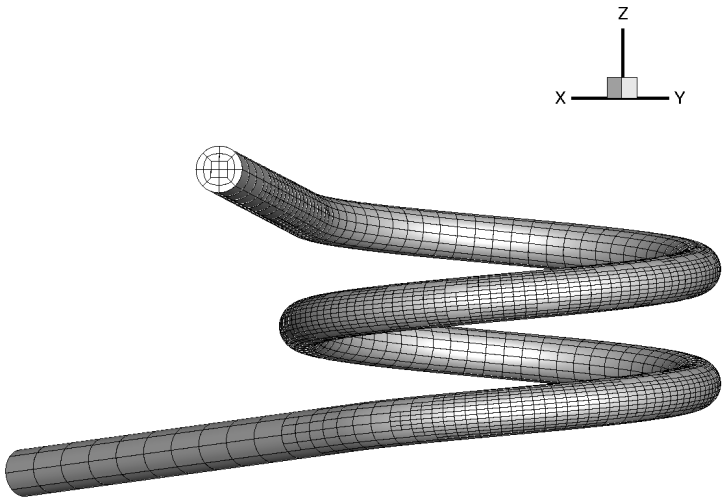
Boundary	condition
Inlet	Hagen-Poiseuille flow $W = 2\left(1 - \left(\frac{2r}{D}\right)^2\right)$
Pipe walls	No slip $u = v = w = 0$
Outlet	Psuedo-traction free $\frac{\mathrm{d}}{\mathrm{d}n} = 0$

Table 3.3: Steady boundary conditions used to simulate steady flows.

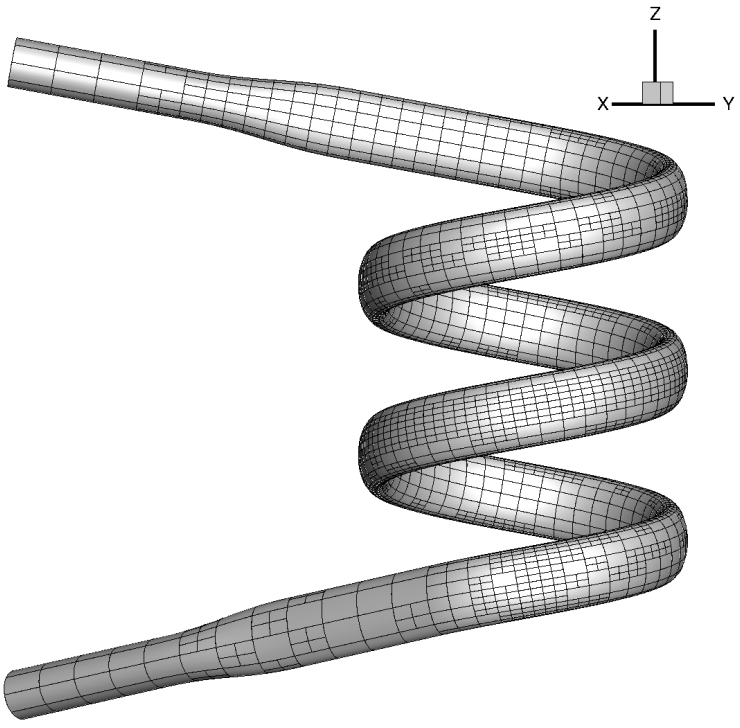
Pipe	MaxTol	MinTol	Number of elements
Small coil	0.0004	0.00004	10329
Large coil	0.0008	0.00008	8761
Unsteady Flow	0.0015	0.00015	18895 at $t = 6.65$

Table 3.4: Tolerances and number of elements used for computations. The time $t = 6.65$ is chosen as it is the last time point shown for the unsteady flows.

To make the fairest comparison between simulated and experimental results, the simulated results are presented at the point where the vector field shows best graphical agreement with the experimental results. This is not necessarily where the x-z viewing plane is coincident with the helical diameter. Figure 3.3 compares the simulated vector field at $y = 0$ and $y = -0.46$. Comparing these results with the experimental result found in figure 3.5f shows that better agreement is found at $y = -0.46$ than at $y = 0$. The dimensional distance between the two planes is 1.8 mm and this is 11% of the helical radius. This is a small change but enough to noticeably affect the vector field. For the large pipe, the best agreement is found at $y = 0.2$.

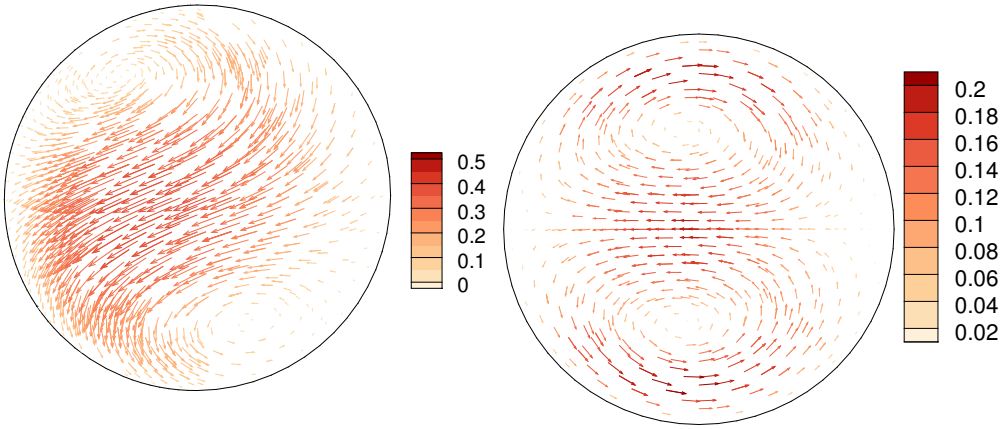


(a) Small pipe



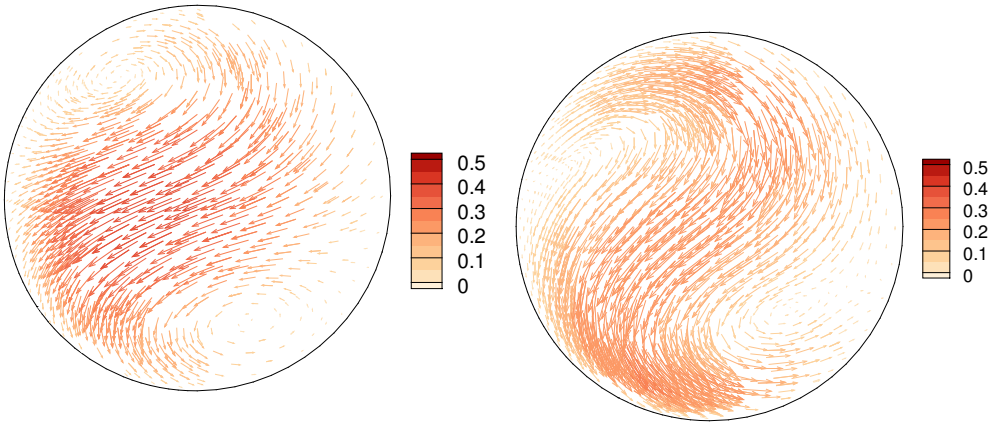
(b) Large pipe

Figure 3.1: Models of the experimental pipes in global Cartesian (x, y, z) space.



(a) u-w (b) U-V

Figure 3.2: Comparison of simulated global u-w cross-flow in the x-z plane at $y = -0.46$ and the U-V cross-flow perpendicular to the centreline at $Z = 55$. The vector magnitude shown in (a) is $\sqrt{u^2 + w^2}$. The vector magnitude shown in (b) is $\sqrt{U^2 + V^2}$.



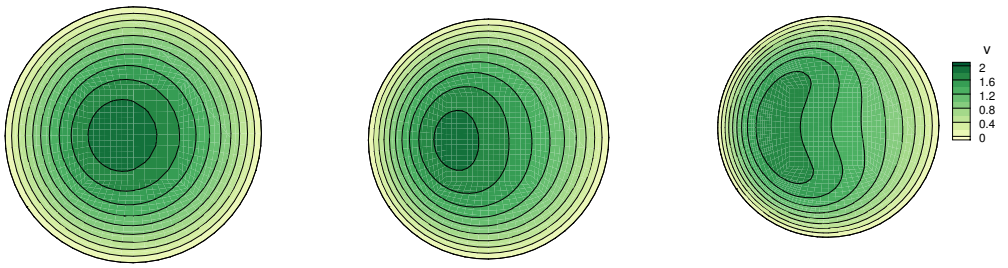
(a) $y = -0.46$ (b) $y = 0$

Figure 3.3: Comparison of simulated global u-w cross-flow in the x-z plane at $y = -0.46$ and the u-w cross-flow in the x-z plane at $y = 0$ for the small pipe at $Re = 150$. The vector magnitude shown is $\sqrt{u^2 + w^2}$.

3.2 Small pipe results

This section compares the experimental results of the smaller pipe from Levy et al. (2015) with the simulated results. The results are presented with the outside wall on the left, the positive y direction in to the page, and with positive z up the page. The flow is moving in to the page and down. This is to present the simulated results in the same manner as Levy et al. (2015).

The simulations for the small pipe are calculated using a maximum tolerance of 4×10^{-4} and a minimum tolerance of 4×10^{-5}. There is no evidence that the chosen tolerance is inadequate. At lower tolerances, there is some evidence that the flow is not fully resolved. The most obvious indicator that the tolerance is insufficient is that flow shows very subtle changes in X-Y cross plane along the centreline of the pipe.



(a) $Re = 40$ (b) $Re = 70$ (c) $Re = 150$

Figure 3.4: Simulated velocity component in global y direction, v at $y = -0.46$ for the small pipe.

The flow shown in figure 3.4 is the flow in the global y direction, v. This provides an approximation of the axial flow, that is, $v \approx W$. The pitch of the pipe is very small, so a global x-z plane is close to a perpendicular plane, and the axial flow is significantly less sensitive to the viewing plane than the cross-flow vector field. The location of maximum magnitude of v shifts towards the outside wall and this shift becomes more pronounced, and the maximum amplitude of the v velocity decreases as the Reynolds number increases.

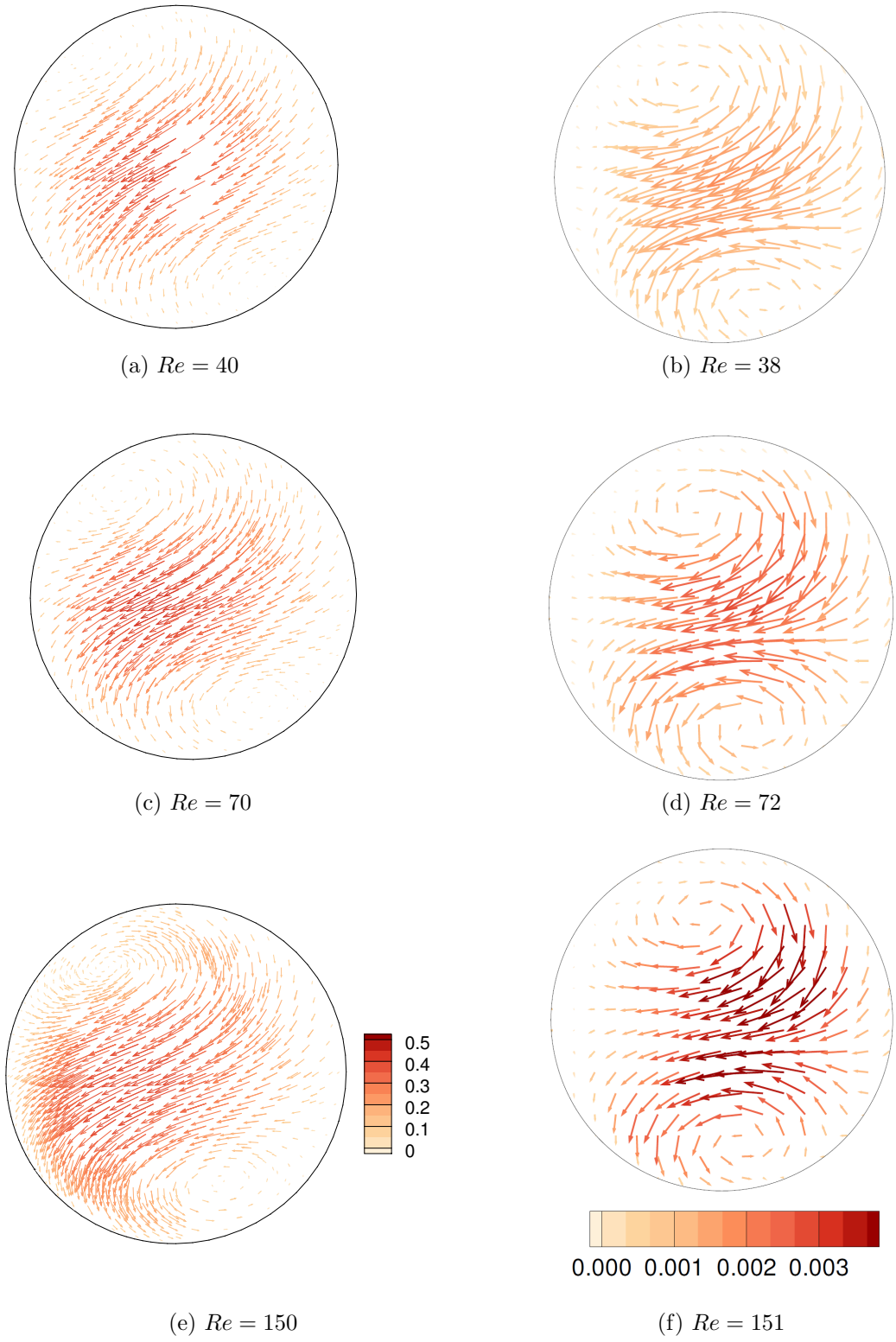
The vector field for both the simulated results and experimental results in figure 3.5 show pairs of counter-rotating vortices. The flow moves towards the outside of the pipe in the middle, and to the inside at the top and bottom. The two vortex centres are not in the middle of the pipe. Instead, the vortex centres are in the upper left and lower right of the pipe. The simulated results show the vortex centres moving closer to the middle of the pipe as

the Reynolds number increases, but this is not observed in the experimental results.

The magnitude for the vectors in the simulated result is two orders of magnitude greater than the experimental results. The experimental vector field is referred to as 'vectors coloured by velocity magnitude' in the caption of figure 3 of Levy et al. (2015), and the average flow velocity is used as a reference quantity to dimensionalise the data. The experimental vector magnitude shown in figure 3.5, in places, is less than both w and w^2, shown in figure 3.6 (b), (d) and (f), so the vector magnitude cannot be $\sqrt{u^2 + w^2}$ or $u^2 + w^2$. The analytic solution from Germano (1989), which underestimates the cross-flow energy at the $Re = 40$ case for this geometry, states that the cross-flow energy E at the centre of this pipe is $E = 0.016$. This is an order of magnitude greater than the experimental value.

Figure 3.6 compares the simulated and experimental global vertical velocity w respectively. The simulated results show the magnitude of the vertical velocity decreases in the middle of the pipe, and the regions of greater magnitude shift towards the corners of the pipe. The experimental results also show the vertical velocity decreasing in magnitude in the middle of the pipe, but there is no growth in magnitude at the corners. Instead there is a small decrease in the experimental maximum magnitude of vertical velocity as the Reynolds number increases. This behaviour is not expected in helical pipe flows.

The vorticity shown for both the simulated results and the experimental results is the component of vorticity in the y direction and is computed from the velocity field. The contours of the simulated vorticity in figure 3.7 (a), (c) and (e), show the magnitude of the vorticity increasing as the Reynolds number increases. In contrast, the experimental results, shown in figure 3.7 (b), (d) and (f), show a small increase from $Re = 38$ to $Re = 72$ and then a noticeable decrease from $Re = 72$ to $Re = 151$. The upper positive region and lower negative region of the vorticity field compare somewhat favourably, but a direct comparison is difficult because the experimental data lies outside the range of the colour bar. The simulated vorticity field has large magnitudes at the top and bottom walls. The experimental grid lacks sufficient resolution to accurately calculate the vorticity in these regions, and the vorticity shown in the experiment is not accurate.

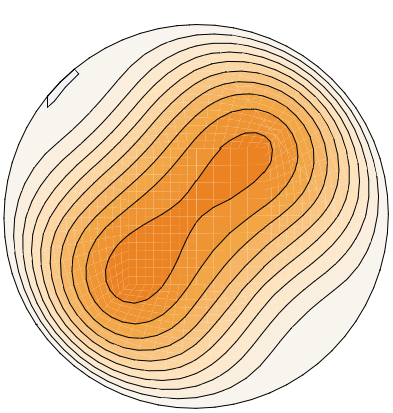


(a) $Re = 40$ (b) $Re = 38$

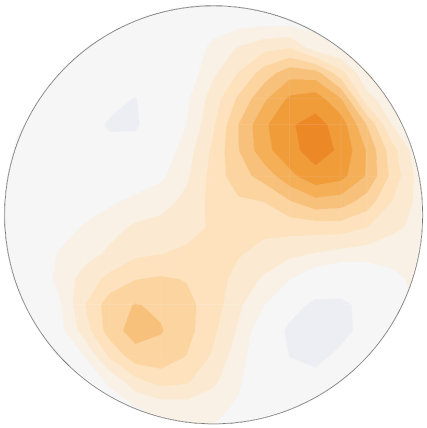
(c) $Re = 70$ (d) $Re = 72$

(e) $Re = 150$ (f) $Re = 151$

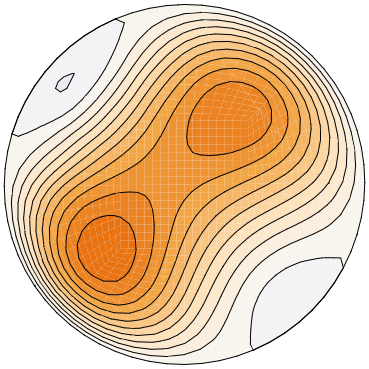
Figure 3.5: Comparison of cross-flow vector field of the small pipe. Figures (a),(c) and (e) show the simulated u-w cross-flow at $y = -0.46$. The vector magnitude shown is $\sqrt{u^2 + w^2}$. Figures (b), (d) and (f) show the experimental cross-flow from Levy et al. (2015).



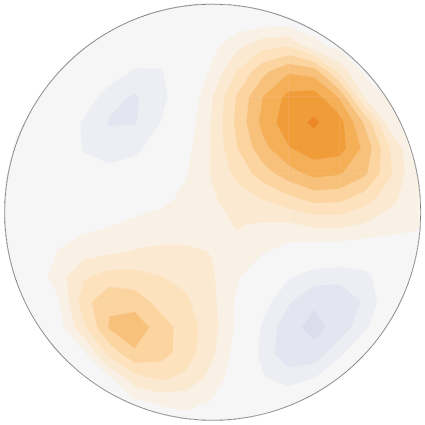
(a) $Re = 40$



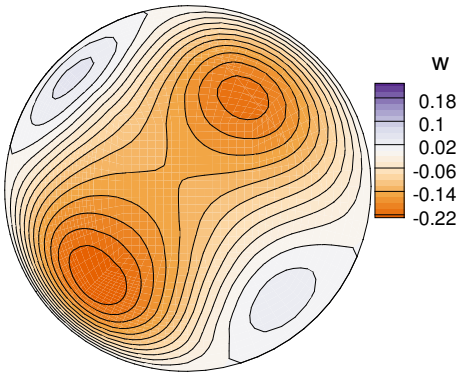
(b) $Re = 38$



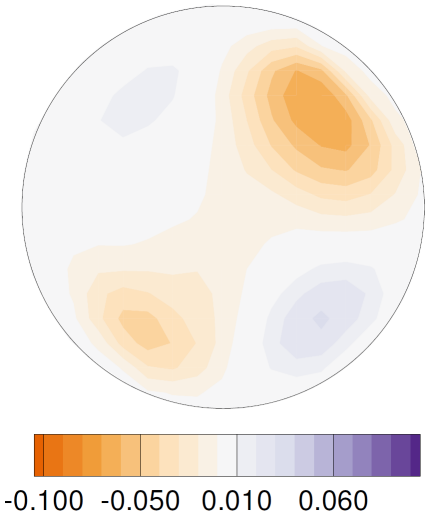
(c) $Re = 70$



(d) $Re = 72$

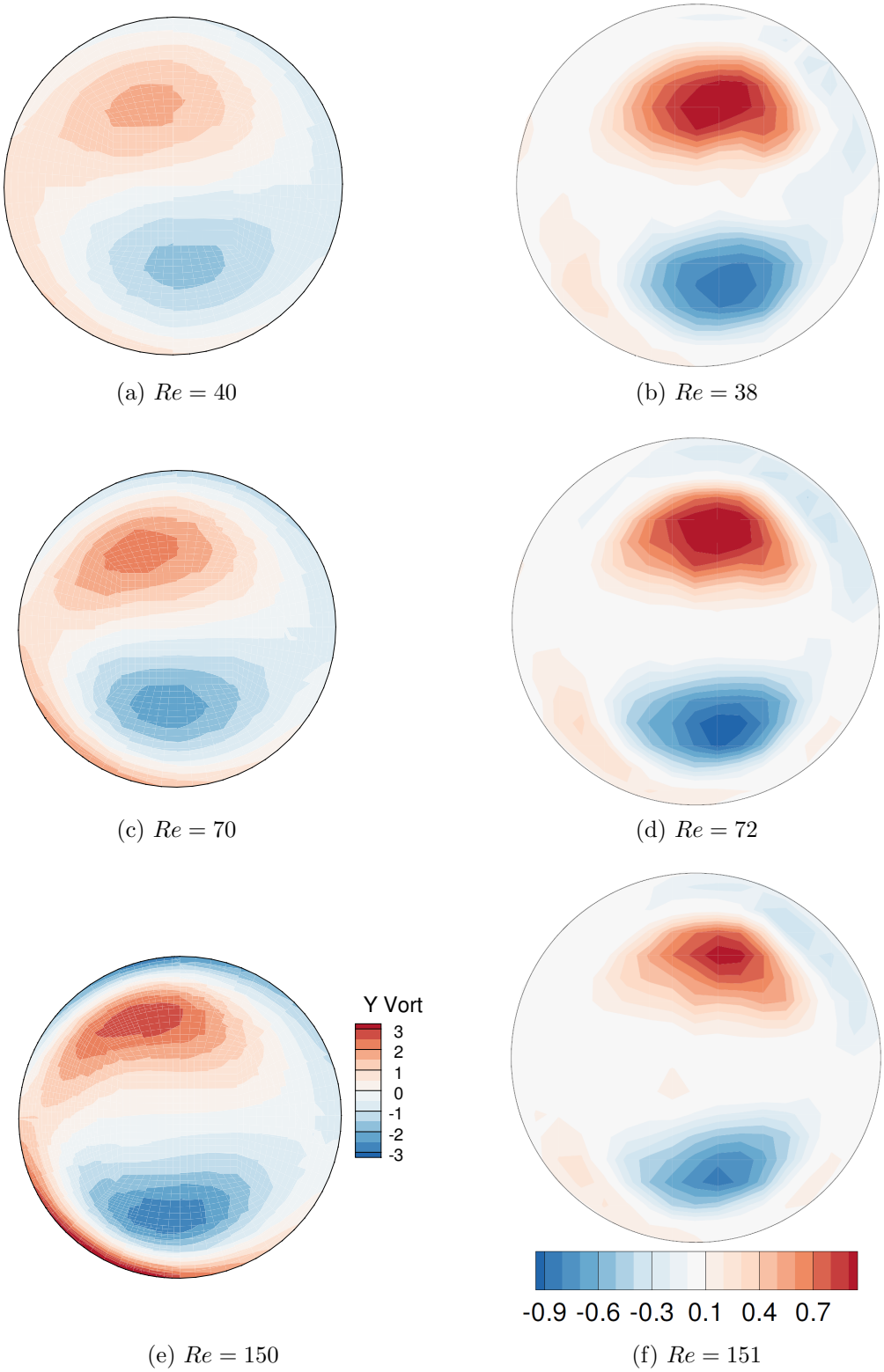


(e) $Re = 150$



(f) $Re = 151$

Figure 3.6: Comparison of velocity in global z direction, w. Figures (a), (c) and (e) show the simulated velocity at $y = -0.46$ and figures (b), (d) and (f) show the experimental results from Levy et al. (2015).



(a) $Re = 40$ (b) $Re = 38$

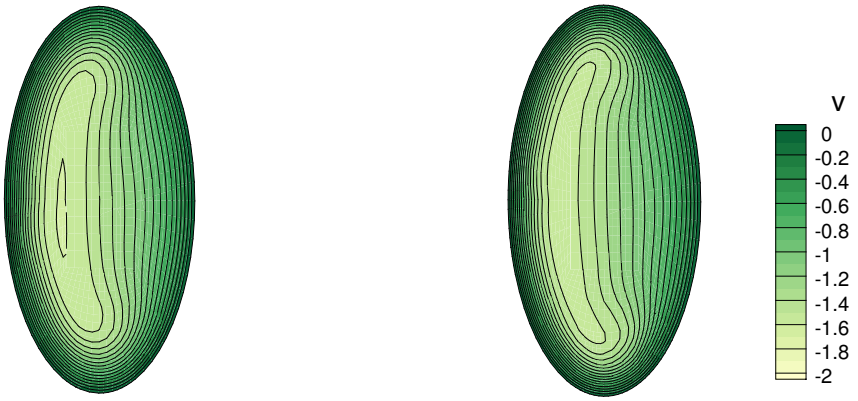
(c) $Re = 70$ (d) $Re = 72$

(e) $Re = 150$ (f) $Re = 151$

Figure 3.7: Comparison of vorticity in y direction. Figures (a), (c) and (e) show the simulated vorticity and figures (b), (d) and (f) show the experimental results from Levy et al. (2015).

3.3 Large pipe results

Similar to the small pipe, the viewing plane for the experimental results is likely a plane that is angled between a plane perpendicular to the pipe centreline and a plane that includes the central helical axis. Good agreement was found by taking an x-z plane at $y = 0.2$. The results for the larger pipe are presented such that the outside pipe wall is on the left of the page and the axial flow is heading out the page and down. The experimental results and simulated results are expected to differ to some degree as the experimental pipe was not exactly elliptical. The inside wall has high in-plane curvature, which results in a smaller corner radius at the top and bottom of the pipe.



(a) $Re = 245$ (b) $Re = 370$

Figure 3.8: Simulated velocity in global y direction, v at $y = 0.2$ for the large pipe.

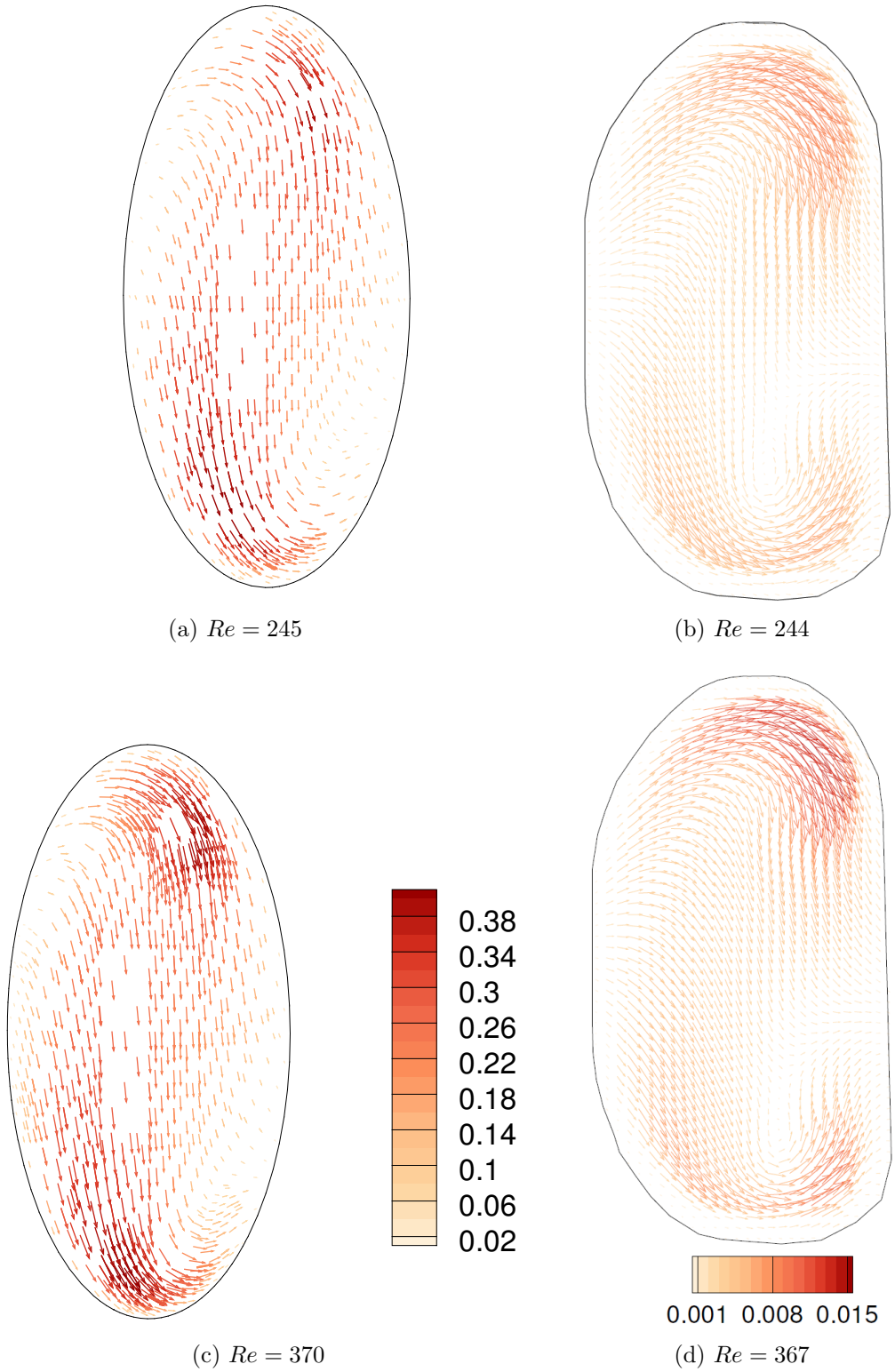
Similar to the smaller pipe, the location of maximum axial velocity shifts to the outside wall, as shown in figure 3.8. This effect is more prominent at $Re = 370$ than at $Re = 245$.

Figure 3.9 shows both the simulated and experimental cross-flow for the large pipe. There are no vortices visible in this plane of orientation. Both the simulated flow and the experimental flow show the cross-flow being directed from the outside wall to the inside wall at the top and bottom of the pipe. The experimental cross-flow moves up the pipe in the lower inside corner, but the simulated flow shows a much smaller vertical magnitude. Figure 3.10 compares the cross-flow in the x-z plane with the cross-flow perpendicular to the centreline. When the cross-section perpendicular to the pipe is considered, there are two counter-rotating vortices present in the vector field of the simulated results. The flow moves from the inside wall to the outside wall in

the centre of the pipe, and from the inside wall to the outside wall at the top and bottom of the pipe. The difference in cross-flows is due to component of the axial flow is the x-z plane, which very noticeably distorts the cross-flow. The magnitude of the experimental vector field is, in some locations, smaller than both w and w^2. A more comprehensive discussion on the magnitude of the experimental cross-flow is given in section 3.2.

Figure 3.11 compares the simulated and experimental component of the velocity in the vertical direction, w. It is difficult to compare the magnitude of the negative regions of w as the minimum vertical velocity is less than the minimum contour level. Despite this, the simulated and experimental magnitudes of w show good agreement in the top right corner of the pipe. Both the simulated and experimental w field show an asymmetry, but this is more pronounced in the experimental results. The magnitudes are higher near the inside wall for both the simulated and experimental results. The regions of higher magnitude in the experimental flow correspond to sharper corners of the pipe wall. Increasing the Reynolds number in this range does not significantly change either the experimental or simulated results. There are small changes in the shape of the contours but the maximum positive and negative values of w are very similar.

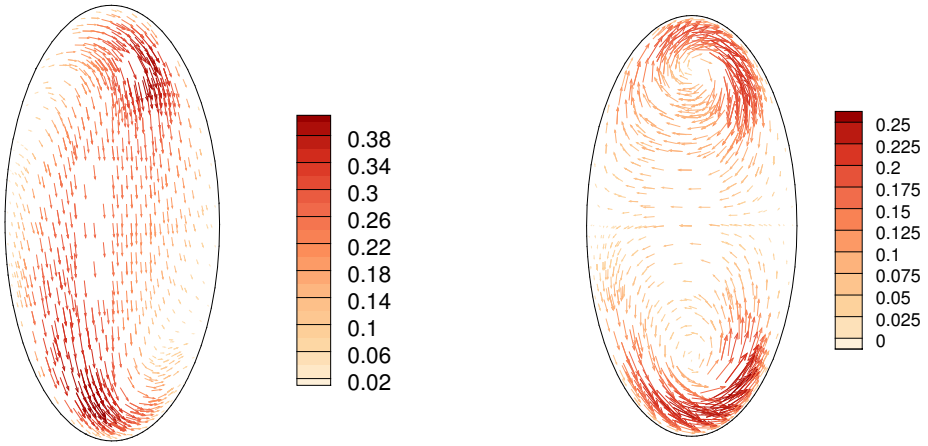
The vorticity shown in figure 3.12 is the component of vorticity in the global y direction, and is calculated from the velocity field for both the experimental and simulated data. The shape of the vorticity contours show good agreement with each other. There is zero vorticity along the pipe wall of the experimental pipe, but, similar to the small pipe, it is possible the vorticity was not accurately calculated as there are fewer points near the wall where the velocity is measured. The contours for the experimental results show large areas where the contour is at a maximum value, implying that the minimum and maximum values of vorticity are beyond the range of the colour bar. The magnitude of the positive section at the top and the negative section at the bottom both show a small increase for both the experimental and simulated results as the Reynolds number increases.



(a) $Re = 245$ (b) $Re = 244$

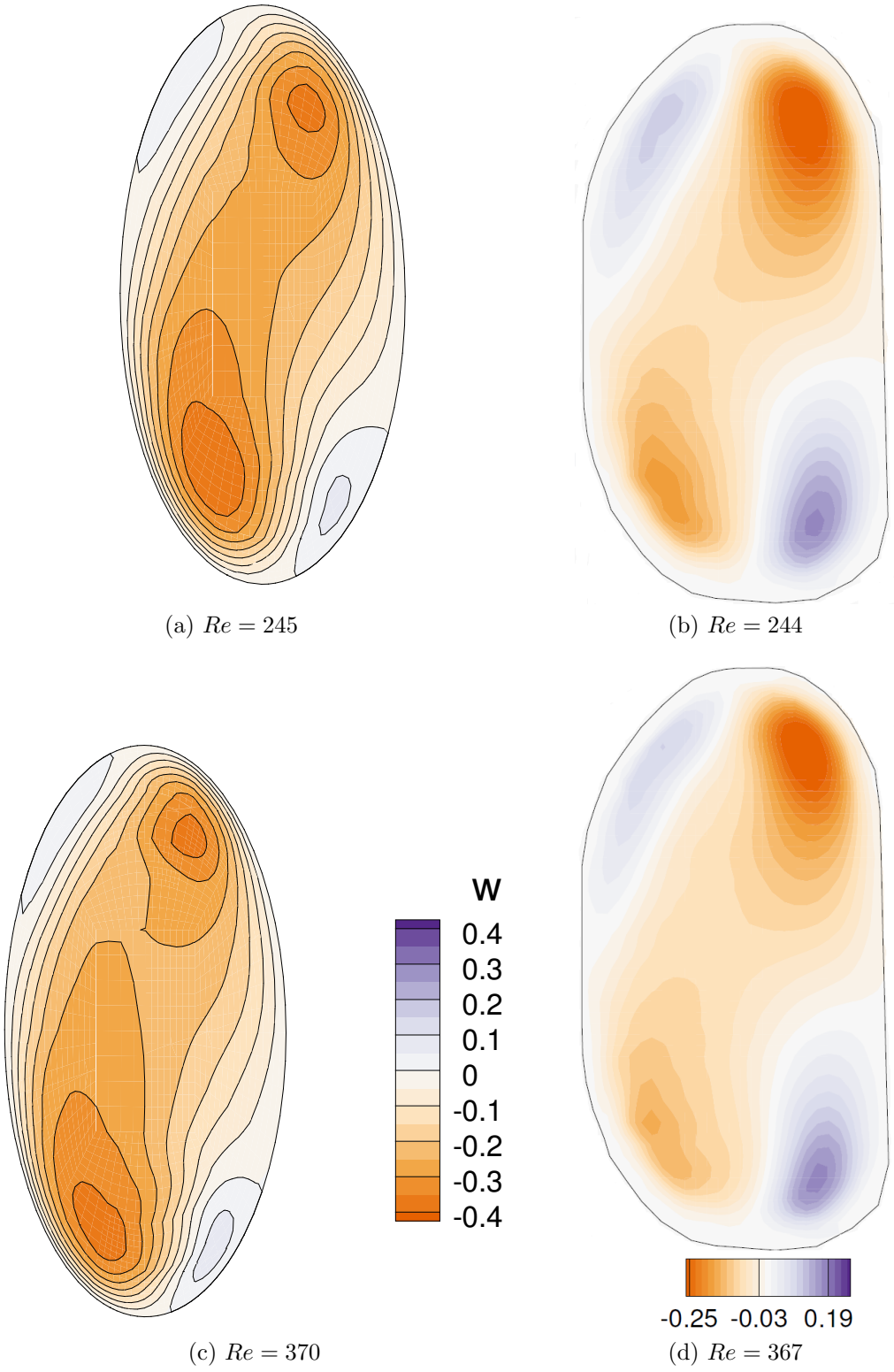
(c) $Re = 370$ (d) $Re = 367$

Figure 3.9: Comparison of cross-flow vector field of the large pipe. Figures (a) and (c) show the simulated u-w cross-flow at $y = 0.5$. The vector magnitude shown is $\sqrt{u^2 + w^2}$. Figures (b) and (d) show the experimental cross-flow from Levy et al. (2015).



(a) u-w (b) U-V

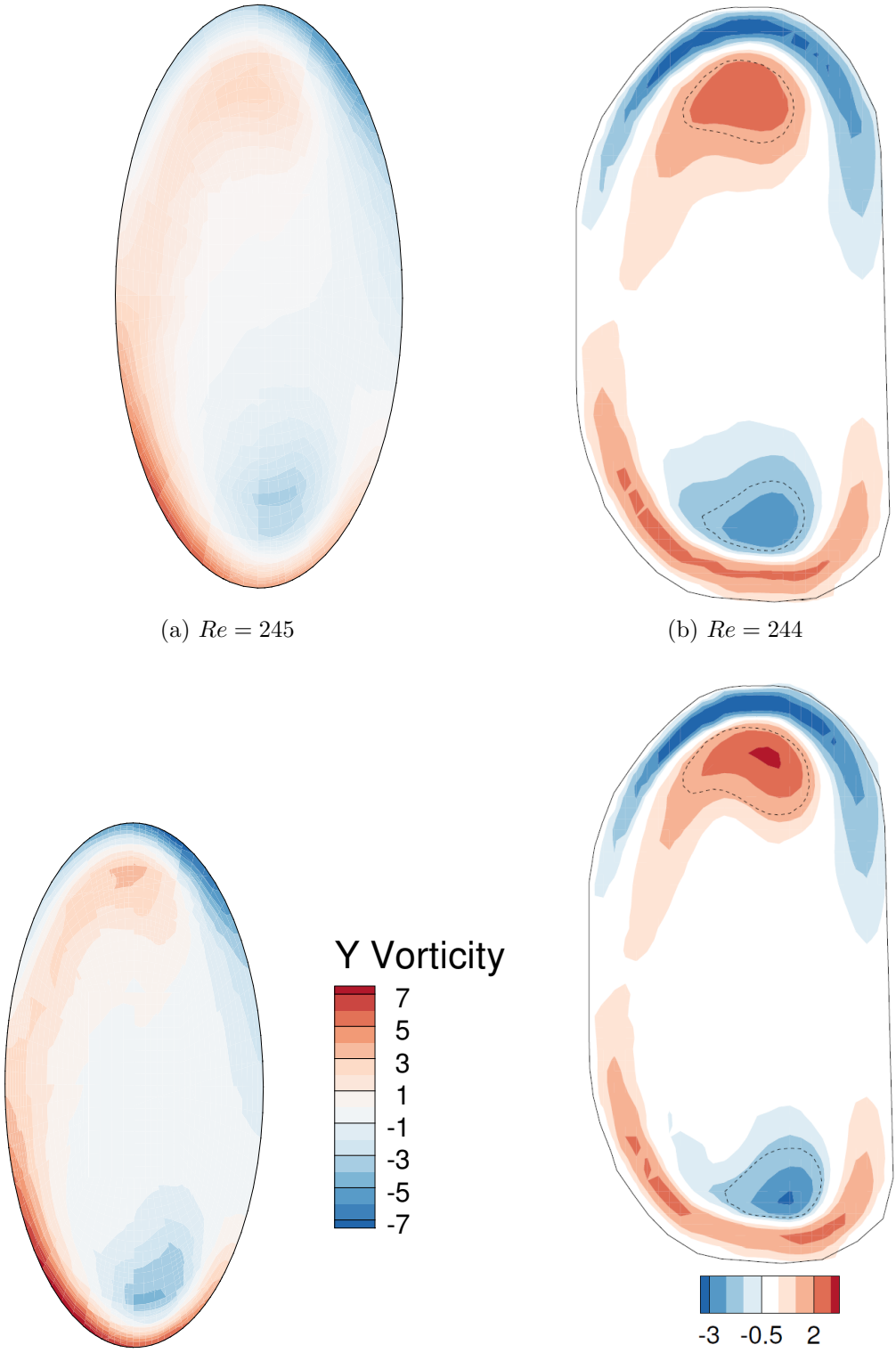
Figure 3.10: Comparison of global u-w cross-flow in the x-z plane at $y = 0.2$ and the U-V cross-flow perpendicular to the centreline at $Z = 55$.



(a) $Re = 245$ (b) $Re = 244$

(c) $Re = 370$ (d) $Re = 367$

Figure 3.11: Comparison of component of velocity in vertical direction, w. (a) and (c) show the simulated results in the plane at $y = 0.5$, and (b) and (d) show the experimental results from Levy et al. (2015).



(a) $Re = 245$ (b) $Re = 244$

(c) $Re = 370$ (d) $Re = 367$

Figure 3.12: Comparison of the simulated and experimental component of vorticity in the y direction. (a) and (c) show the simulated vorticity, and (b) and (d) show the experimental results from Levy et al. (2015).

3.4 Stopping flow

Levy et al. (2015) consider the case where steady flow is suddenly stopped in the large pipe. The cross-flow is measured for a length of time after the flow is stopped. These conditions are simulated in `oomph-lib` to compare the experimental results with simulated results. The unsteady solver is used with the steady inlet condition from section 3.1, table 3.3 with an initial condition of $(u, v, w) = \mathbf{0}$ and a Reynolds number of $Re = 490$. The simulation is run until a steady state is achieved, and then stopped. Stopping the flow in one time step proved to be too computationally intensive to solve using the unsteady solver. The sudden stop creates very sharp temporal and spatial gradients. The adaptive mesh refinement tries to resolve these sharp spatial gradients, which leads to very high element counts and yields a system of equations in too many unknowns to be computationally tractable. To compensate for this, a period of time is introduced where the flow is slowed down to a complete stop over a number of time steps. The inlet smoothly ramps down using the function

$$G(t) = \begin{cases} 1 & t \leq t_{\text{fall}} \\ 1 - \left[6 \left(\frac{t - t_{\text{stop}}}{t_{\text{fall}}} \right)^5 - 15 \left(\frac{t - t_{\text{stop}}}{t_{\text{fall}}} \right)^4 + 10 \left(\frac{t - t_{\text{stop}}}{t_{\text{fall}}} \right)^3 \right] & t_{\text{stop}} < t \leq t_{\text{stop}} + t_{\text{fall}} \\ 0 & t_{\text{stop}} + t_{\text{fall}} < t. \end{cases} \tag{3.1}$$

The inlet for the unsteady case is then

$$W(r, t) = 2G(t) \left[1 - \left(\frac{r}{R} \right)^2 \right].$$

The results shown below use a value of $t_{\text{stop}} = 5.5$ and $t_{\text{fall}} = 0.5$, which corresponds to 20 time steps.

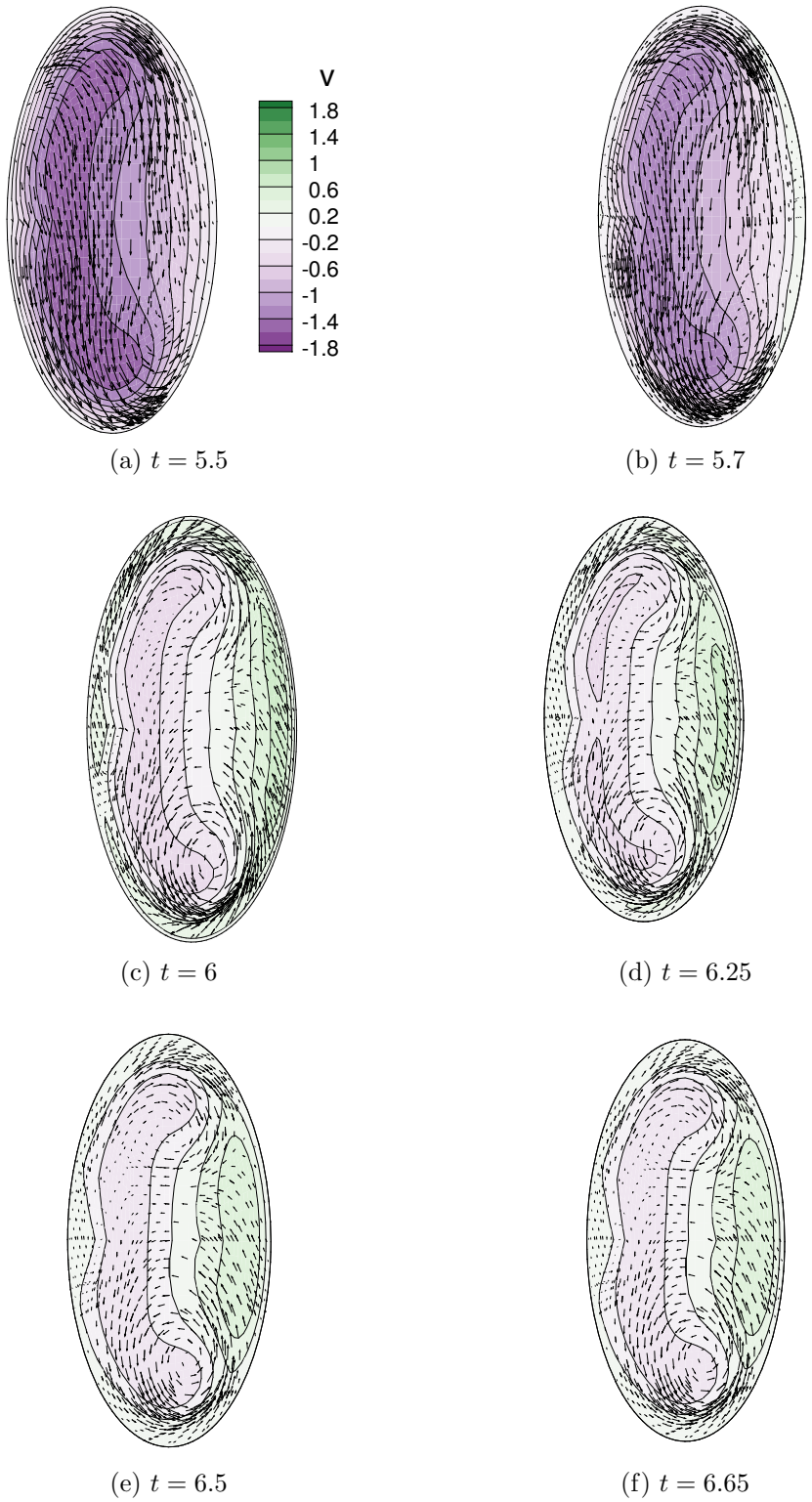
The simulated and experimental results, shown in figures 3.13 and 3.14 respectively, show that as the flow stops and after the flow has stopped, the cross-flow becomes less intense. The simulated flow is shown at the same perpendicular x-z slice as the steady results for the large pipe in section 3.3. Figure 3.15 compares the experimental steady cross-flow and cross-flow as the flow is suddenly stopped, which differ slightly as they are from different experiments. Figure 3.15a shows that the steady cross-flow vector field at $Re = 490$ has a very similar pattern to the cross-flow vector field at $Re = 367$, shown in figure 3.9d. Figure 3.15b shows two counter-rotating vortices, which is more similar to the cross-flow perpendicular to the pipe centreline in figure 3.10b. It is possible that the plane used to take results in the suddenly

stopping case was more closely aligned to the plane perpendicular to the pipe centreline.

Reverse axial flow is seen in the simulated results, as seen by the green regions in figure 3.13. This reverse flow is seen in Jewell & Denier (2013) when suddenly stopping flow was investigated in a straight pipe. They found a radially symmetric region of counter-flow outside a radially symmetric region of downstream flow in the pipe centre. In the helical pipe, the downstream axial flow shifts towards the outside wall and the reverse flow is stronger closer to the inside wall, but is still present on the outside wall. As the flow slows down, the cross-flow becomes less intense and the two counter-rotating vortices are more apparent in the y slice. The magnitude of the axial flow is decreasing in time, and hence the contribution of the axial flow in the x-z cross-flow is smaller. This leads to the normal and binormal velocities dominating the cross-flow, which leads to two counter-rotating vortices of roughly equal size, as seen in figure 3.2b.

3.5 Conclusion

In this chapter, computational models of the experimental pipes from Levy et al. (2015) were created. The results from the simulations were graphically compared with the experiment results. The shapes of the vector fields, vertical velocity and vorticity show good agreement, although there was difficulty in interpreting some of the experimental results as data went beyond colour bars, and the experimental vector magnitude could not be determined. When the flow is suddenly stopped, both the experimental and simulated results show two counter-rotating vortices that become weaker in time. The best agreement between simulated and experimental results was found by considering the orientation and alignment of the plane when viewing the simulated results. Small shifts in alignment of the plane lead to very noticeable changes in the cross-flow. To measure the true cross-flow, experimentalists must take great care to align the measurement plane to be perpendicular to the pipe centreline.

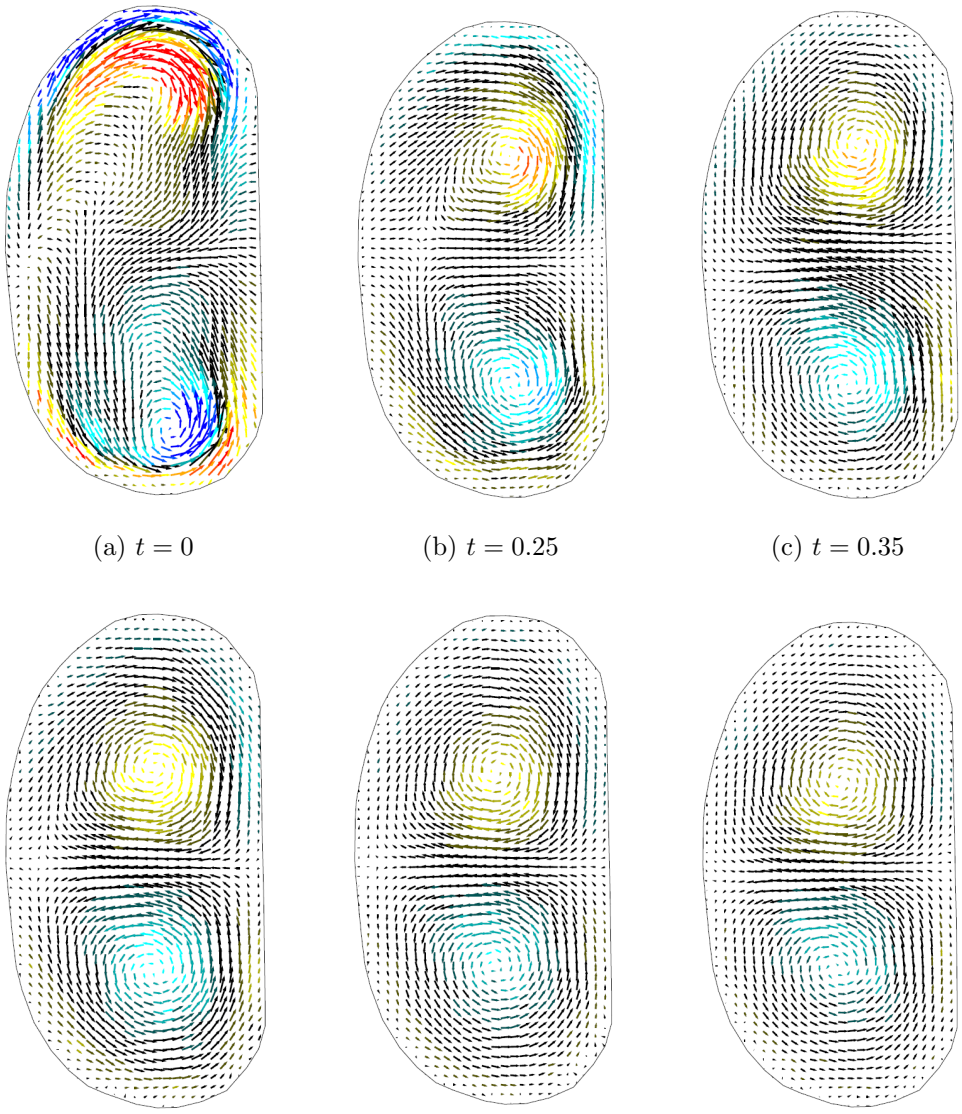


(a) $t = 5.5$ (b) $t = 5.7$

(c) $t = 6$ (d) $t = 6.25$

(e) $t = 6.5$ (f) $t = 6.65$

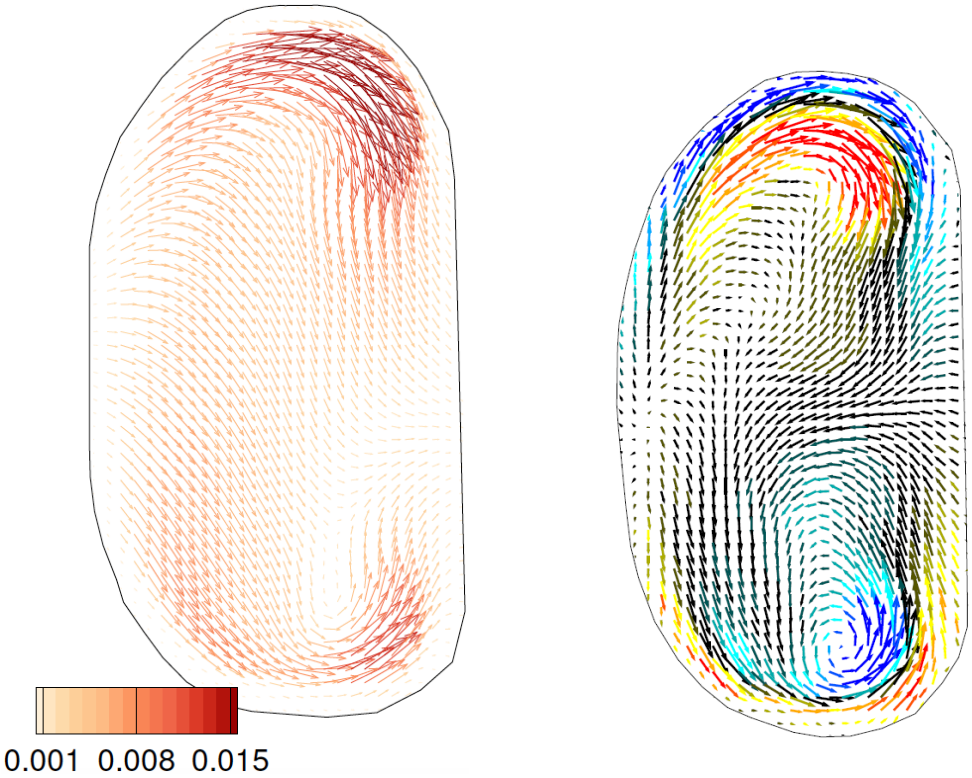
Figure 3.13: Simulated velocity in global y direction, v and $x - z$ plane cross-flow for suddenly stopping flow at $y = 0.2$, $Re = 490$ for the large experimental pipe. The outside wall is on the left of the page.



(a) $t = 0$ (b) $t = 0.25$ (c) $t = 0.35$

(d) $t = 0.45$ (e) $t = 0.55$ (f) $t = 0.65$

Figure 3.14: Experimental results for suddenly stopping flow at $Re = 490$, where t is time after the stop, for the large experimental pipe (Levy et al. 2015). The colour shows vorticity magnitude.



(a) steady (b) unsteady $t = 0$

Figure 3.15: Comparison of experimental steady vector field and vector field when flow is suddenly stopped for $Re = 490$ (Levy et al. 2015).

Chapter 4

A model of flow through umbilical vascaluture

Flows through both umbilical veins and arteries are approximated in this chapter by simulating flows through helical pipes. The flows through umbilical veins are approximately steady, but the flows through arteries are pulsatile, due to the forcing of the fetal heart. For the purposes of this research, three umbilical cord models from Wilke (2016) have been chosen. The U_{c7} model, shown in figure 4.1, is hypocoiled, the U_{t3} model, shown in figure 4.2, is normocoiled, and the U_{c2} model, shown in figure 4.3 is hypercoiled. The non-dimensional parameters that define the pipe centreline are shown in table 4.2. The effect of the aspect ratio, e, and centreline geometry, on axial flow, cross-sectional flow and pressure are investigated for both steady and unsteady flows.

A survey of the umbilical parameters for the vein and artery was performed by Wilke (2016). A summary of this survey is found in table 4.1. This survey found that using $Re = 200$ is suitable for modelling both arteries and veins, and $St = 2/41$ is suitable for modelling the pulsatile flow in arteries. It was also found that the temporal variation of the centreline velocity has a saw-tooth profile.

	min	max	nominal
Arterial diameter, $\hat{D}_A$ (10^{-3} m)	2.8	6.8	4
Venous diameter, $\hat{D}_V$ (10^{-3} m)	5.8	15	7
Arterial flow-rate, $\hat{Q}_A$ (10^{-6} m^3/s)	1.00	4.21	2.4
Arterial flow-rate, $\hat{Q}_A$ (10^{-6} m^3/s)	1.45	8.75	4.4
Blood viscosity, $\hat{\nu}$ (10^{-6} m^2/s)	3.40	4.72	3.77
Fetal heart-rate, f (bpm)	110	160	140

Table 4.1: Summary of umbilical cord parameters from Wilke (2016).

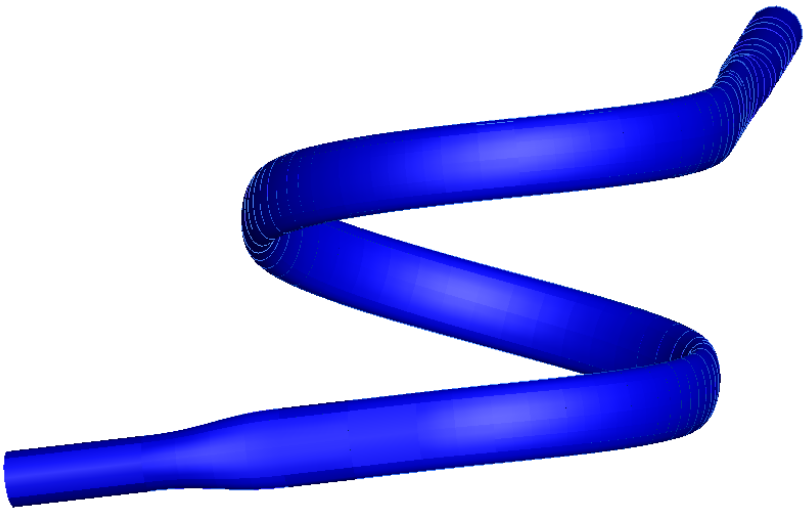


Figure 4.1: Hypocoiled pipe U_{c7} at $e = 0.5$.

In this research, the average pressure gradient in the helical section is used as the primary measure of pressure in the pipe. Wilke (2016) pointed out that one of the largest influences on the pressure drop (ΔP) between the inlet and outlet is the length of the pipe. The pressure gradient is independent of length. The lengths of the pipe models in this chapter are not representative of typical of umbilical cords, so changes in pressure gradient better reflect changes in the loading of the fetal heart. The pressure gradient is increased in the cases of both $e = 0.5$ and $e = 2$ for all centrelines. For centrelines U_{c7} and U_{t3}, the pressure gradients is highest in magnitude for $e = 2$. The pressure gradient is highest for $e = 0.5$ for the U_{c2} geometry.

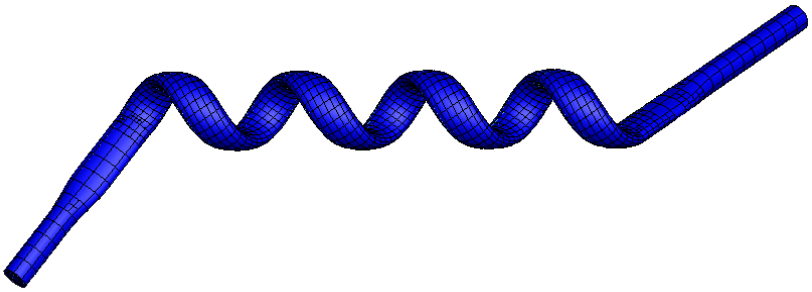


Figure 4.2: Normocoiled pipe U_{t3} at $e = 0.5$.

	R_c	H	κ	τ	λ	η	γ	L	coil length
U_{c7}	4	4.74	0.24	0.046	0.12	0.023	0.0080	50	25.6
U_{t3}	1.10	4.66	0.63	0.42	0.32	0.21	0.035	50	8.3
U_{c2}	0.25	4.73	0.40	1.20	0.2	0.60	0.14	50	5.0

Table 4.2: Geometric and dynamic non-dimensional parameters for the three umbilical models. The parameters λ, η and γ are included for comparisons with Liu & Masliyah (1993). The value for γ is for $Re = 200$.

4.1 Steady flow

The steady results are used to investigate the effect of changing the eccentricity, e, in umbilical veins. The results in section 4.1.2 show that, for most cases, the effect of e on flow development is minimal and changing e does not fundamentally change the developed flow. Remarkably, for pipe U_{c2} at $e < 1$ and sufficiently high Reynolds number, the flow does not fully develop. Instead, it takes on a behaviour that is periodic in length. Changing e also changes the limiting γ that determines whether there is one or two vortices in the cross-flow. The pipes are presented in order of ascending torsion so the results most similar to curved pipes are shown first and the discussion on the effect of torsion on the flow can follow logically. The tolerances and number of elements used to compute each solution at $Re = 200$ are shown in table 4.4. As the torsion of the pipe increases, more elements are needed to resolve the flow.

The boundary conditions used to model the steady flow are the same conditions used to model the steady flow in section 3.1, and are found in table 3.3.

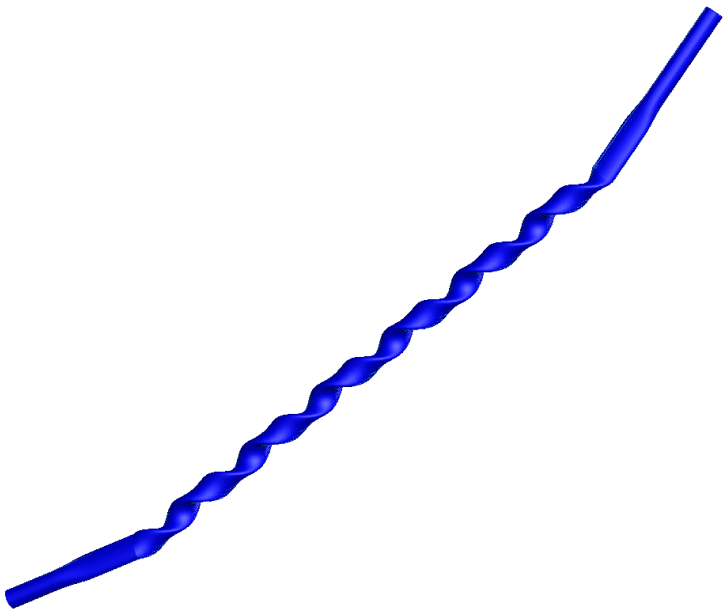


Figure 4.3: Hypercoiled pipe U_{c2} at $e = 0.5$.

	Z_{i}	Z_{i1}	Z_{i2}	Z_{h}	Z_{o}	Z_{o1}	Z_{o2}	Z_{e}
Regular	0	2.5	5	7.5	40	42.5	45	50
Long	0	4	8	12	64	68	72	80

Table 4.3: Important pipe lengths for the three umbilical models. Most simulations are performed with the regular pipe length. The long pipe length is used to further investigate the U_{c2} $e = 0.5$ case and its usage is specified.

4.1.1 Flow development

This section investigates the effect of the sudden change in curvature, where the pipe centreline changes from straight to helical, on the development of the flow. The discontinuity in curvature creates a strong cross-flow and distortion in the axial flow. In most cases, the flow becomes fully developed downstream from this change and the axial and cross-flow become axially invariant. Figures 4.4 to 4.6 show the cross-sectionally averaged cross-flow energy defined in section 2.6 along the length of the pipe. The cross-sectionally averaged cross-sectional energy $\bar{E}$ for each flow becomes axially invariant except for the case of the U_{c2} centreline when $e = 0.5$. This flow develops a structure that is periodic in length, and this behaviour will be discussed further in section 4.1.3. For each flow, there is a sharp increase in $\bar{E}$ at $Z = 7.5$, which

	e	MaxTol	MinTol	Number of elements
U_{c7}	$e = 0.5$	0.0008	0.00008	8110
	$e = 1$	0.0008	0.00008	7599
	$e = 2$	0.0008	0.00008	7543
U_{t3}	$e = 0.5$	0.0008	0.00008	14228
	$e = 1$	0.0008	0.00008	10147
	$e = 2$	0.0008	0.00008	11782
U_{c2}	$e = 0.5$	0.0006	0.00006	28466
	$e = 1$	0.0006	0.00006	22110
	$e = 2$	0.0008	0.00008	17049

Table 4.4: Tolerances and number of elements used for computations.

is the point of transition from a straight pipe to a helical pipe. Some flows, such as U_{t3} where $e = 2$, show a behaviour that is similar to a damped oscillation before becoming fully developed. For pipes with $e \neq 1$ there is a small increase at $Z = 2.5$, where the cross-section changes from circular to elliptical. The cross-flow energy caused by the change in cross-section dissipates before the change in centreline.

Figure 4.7 shows the cross-flow development for the U_{t3} centreline when $e = 2$. The shape of the cross-flow develops quickly. The shape of the cross-flow does not significantly change from $Z = 10$ to $Z = 20$. Figures 4.7b to 4.7f all show one large vortex at the bottom of the pipe and a layer of inwards pointing cross-flow on top of this. The magnitude of the cross-flow develops over a longer length. There is a noticeable decrease in magnitude from $Z = 10$ to $Z = 12$, but a very small change from $Z = 18$ to $Z = 20$. This agrees with the plot in figure 4.5.

Figure 4.8 shows the axial flow development for the U_{t3} centreline when $e = 2$. The location of the maximum magnitude of axial velocity shifts to the outside of the pipe at $Z = 10$. After shifting back to the centre of the pipe at $Z = 12$, it shifts back to the outside of the pipe at $Z = 15$. There is a small change in the shape of the contours from $Z = 15$ to $Z = 18$. There is almost no change between $Z = 18$ to $Z = 20$. The flow is considered fully developed at $Z = 20$ for this case as there is a less than 2% change in $\bar{E}$. Between $Z = 20$ and $Z = 39$, there is no graphical change in either the axial flow or cross-flow. The flow is found to become fully developed for all cases except for U_{c2} when $e = 0.5$.

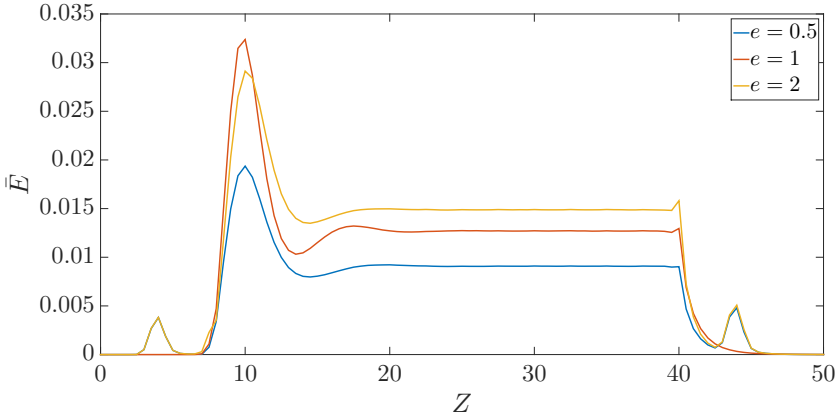


Figure 4.4: Cross-sectionally averaged cross-sectional energy along the length of pipe for U_{c7} at $Re = 200$.

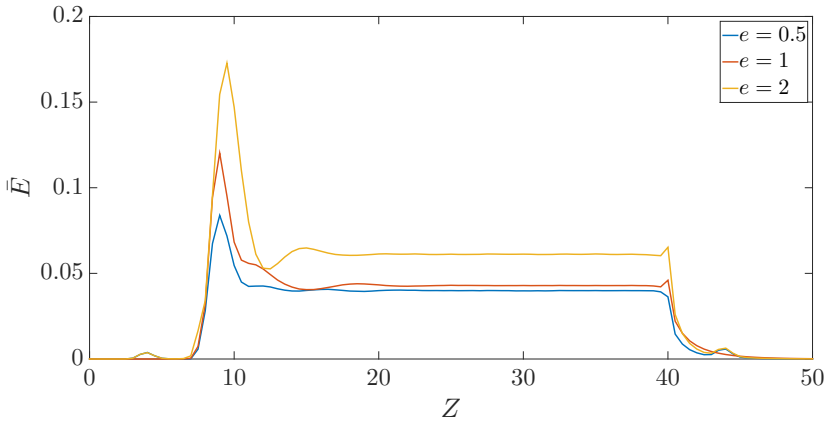


Figure 4.5: Cross-sectionally averaged cross-sectional energy along the length of pipe for U_{t3} at $Re = 200$.

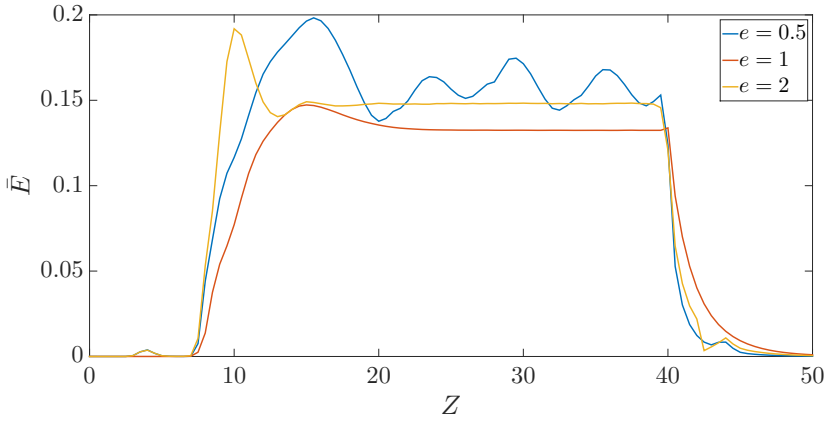
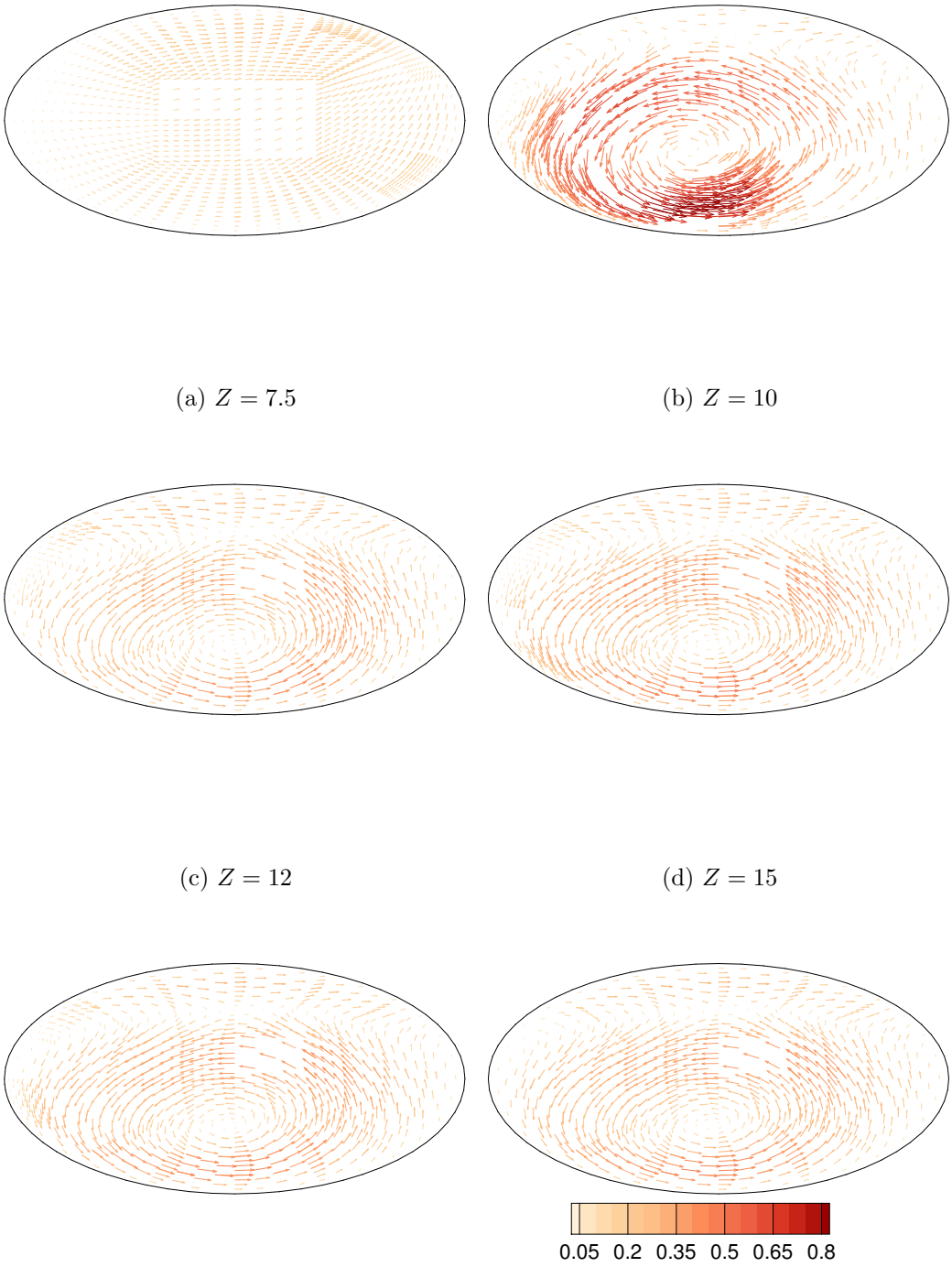


Figure 4.6: Cross-sectionally averaged cross-sectional energy along the length of pipe for U_{c2} at $Re = 200$.

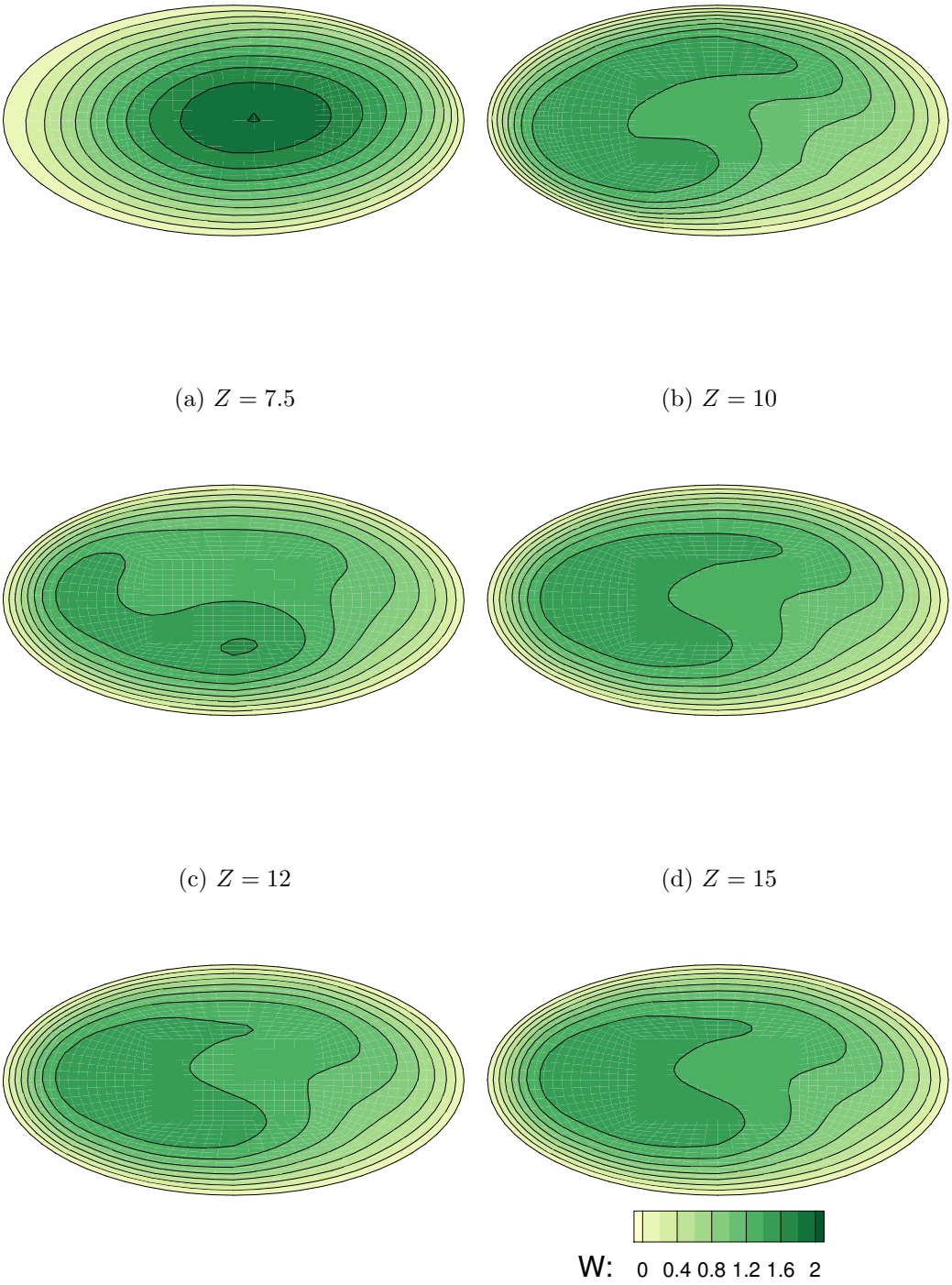


(a) $Z = 7.5$ (b) $Z = 10$

(c) $Z = 12$ (d) $Z = 15$

(e) $Z = 18$ (f) $Z = 20$

Figure 4.7: Cross-flow development for U_{t3} $e = 2$ at $Re = 200$ between $Z = 7.5$ and $Z = 20$. The contour show the vector magnitude $\sqrt{U^2 + V^2}$. The outside wall of the pipe is on the left of the page.



(a) $Z = 7.5$ (b) $Z = 10$

(c) $Z = 12$ (d) $Z = 15$

(e) $Z = 18$ (f) $Z = 20$

Figure 4.8: Axial flow development for U_{t3} $e = 2$ at $Re = 200$ between $Z = 7.5$ and $Z = 20$. The outside wall of the pipe is on the left of the page.

4.1.2 Developed flow features

For most of the modelled flows, the flow becomes fully developed. This section demonstrates the effect of centreline parameters and aspect ratio on the fully developed axial flow and cross-flow. Comparisons with Liu & Masliyah (1993) show that the parameter $\gamma = \eta/\left(Re^{3/2}\lambda^{3/4}\right)$ is found to accurately predict the behaviour of both the axial and cross-flows, even for pipes of very different Reynolds numbers, curvature and torsion. Changing the aspect ratio of the cross-section changes the limiting γ that determines whether there is one cross-flow vortex or two vortices.

Figure 4.9 compares the fully developed computed results for each pipe at $e = 1$ with results from Liu & Masliyah (1993) that have a similar γ. The location of the maximum magnitude of axial velocity shifts to the outside wall for pipes U_{c7} and U_{t3}, which both have small γ. This is similar to the axial results in chapter 3, in figure 3.4. For U_{c2}, the location of the maximum magnitude of axial velocity shifts to the inside wall. Increasing γ further at the same Reynolds number and curvature will result in the axial flow approaching Pouisielle flow (Liu & Masliyah 1993).

Figure 4.10 shows the fully developed axial flow for U_{t3} for $e = 0.5$, $e = 1$ and $e = 2$. The effect of e on the axial flow is minimal, and the location of the maximum magnitude of axial velocity is close to the outside wall in each case. The effect of e on the axial flow is small for all pipes. Changing e does not fundamentally change the axial flow for any pipe centreline.

Figure 4.11 compares the fully developed cross-flow vector field for the simulated results at $e = 1$ and for flows of similar γ from Liu & Masliyah (1993). Despite being for a different Reynolds number and curvature, which leads to the cross-flow vectors having a significantly smaller magnitude, the cross-flow pattern is remarkably similar. Liu & Masliyah (1993) find that, for $e = 1$ and $\gamma = 0.15$, the cross-flow vector field changes from having two vortices to one. The computed results change at a γ between $\gamma = 0.035$ and $\gamma = 0.14$, as the results show two cross-flow vortices at $\gamma = 0.035$ and one vortex at $\gamma = 0.14$.

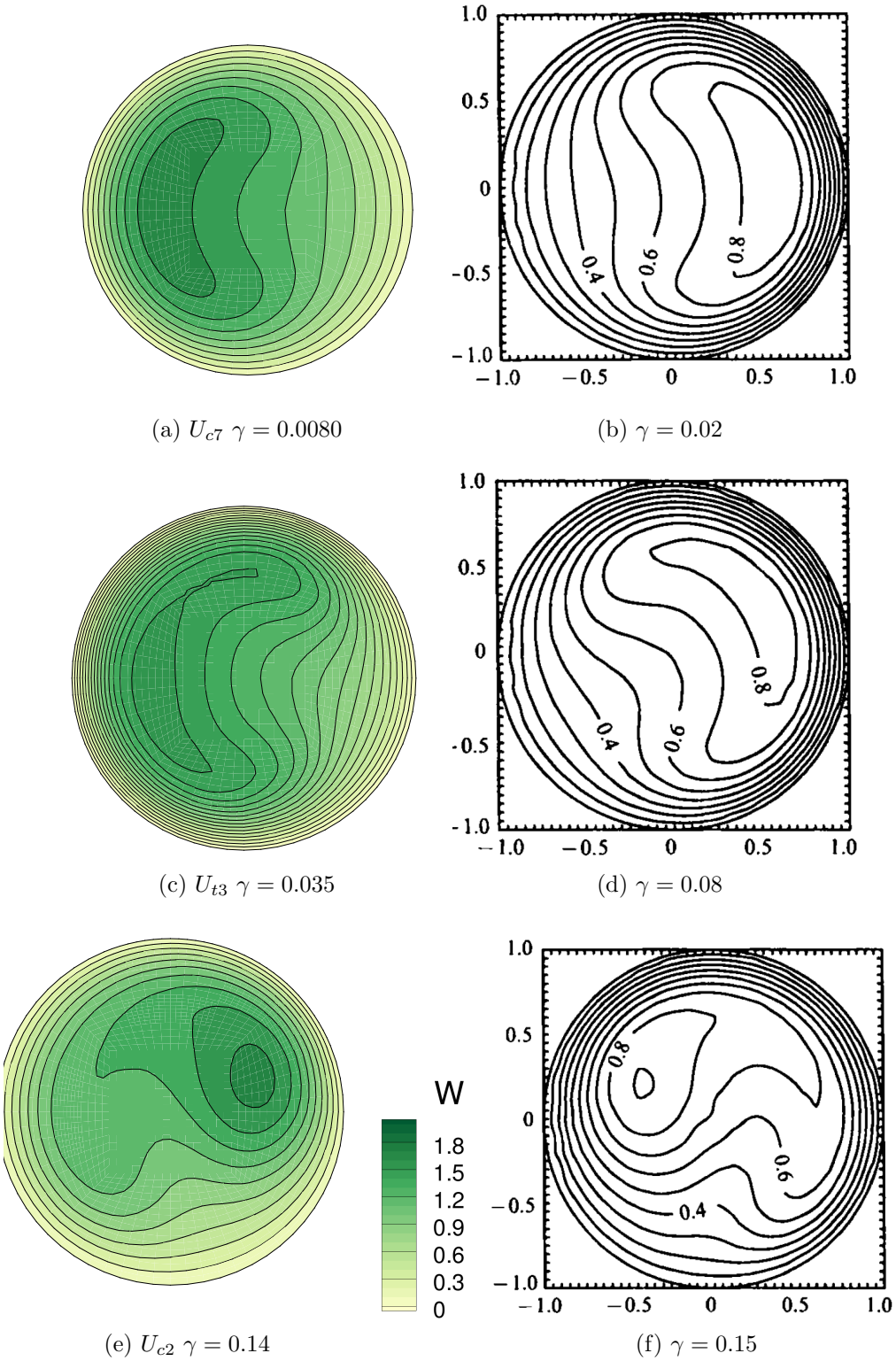
Figures 4.12 to 4.14 show the fully developed cross-flow vector field for each pipe. Figure 4.12 shows that, for pipe U_{c7}, all cases of e show that the flow moves to the outside wall in the middle of the pipe and to the inside at the top and bottom of the pipe and the vortex centres are in the middle of the pipe. For $e = 0.5$, $e = 1$ and $e = 2$, there are two counter-rotating vortices of equal size. For $e = 2$, the vortex centres shift towards the outside wall of the pipe in a way that is asymmetrical.

Figure 4.13 shows the fully developed cross-flow for the U_{t3} pipe. The $e = 0.5$ and $e = 1$ cases show two counter-rotating vortices that rotate in

the same direction as those for U_{c7}, and the bottom vortex is larger than the top vortex. For $e = 1$, the flow through the middle is towards the top outside corner of the pipe. This is not evident for the $e = 0.5$ case, where the flow through the middle is only horizontal. Only one vortex is visible for the $e = 2$ case, with the vortex centre towards the bottom of the pipe. For this flow, $\gamma = 0.035$, so the value of γ at which the cross-flow transitions from two vortices to one vortex is significantly smaller for $e = 2$. The value of γ where the flow transitions from two vortices to one vortex occurs is in the range of $\gamma = 0.0080$ and $\gamma = 0.035$. The possible values of γ are significantly smaller that $\gamma = 0.15$, which is the value of γ where the flow structure changes for a circular cross-section (Liu & Masliyah 1993).

Figure 4.14 shows the fully developed cross-flow vector field for pipe U_{c2} for $e = 1$ and $e = 2$. Both flows show one large vortex with the flow going from top to bottom on the inside of the pipe and from bottom to top on the outside of the pipe. The vortex centre moves to the inside of the pipe. For $e = 1$, increasing the torsion of the pipe further will lead to counte- rotating vortices that are side by side, with the flow going up in the middle of the pipe, and down at the inside and outside walls (Liu & Masliyah 1993). The value of γ at which the cross-flow transitions from two vortices to one vortex is strongly affected by e, so the effect of increasing torsion when $e \neq 1$ is unknown.

Figure 4.15 compares the cross-flow energy contours at $Re = 100$ and $Re = 200$ for the U_{c7} pipe at $e = 1$. At $Re = 200$, the maximum cross-flow energy is larger, but the energy is more sparse in the middle of the pipe and more concentrated at the top and bottom of the pipe compared to $Re = 100$. This behaviour causes the cross-sectionally averaged area $\bar{E}$ to decrease. A similar behaviour is seen in Liu & Masliyah (1993), where the cross-flow patterns for a fixed helical pipe geometry and $e = 1$ are shown at a range of Reynolds numbers. Increasing Dn for a pipe of $\lambda = 0.25$ and $\eta = 0.020$ leads to the cross-flow becoming more concentrated in the top and bottom of the pipe. Figure 4.16 shows $\bar{E}$ over the length of the pipe for various Reynolds numbers for $e = 0.5$. At $Re = 150$ and $Re = 200$, the fully developed $\bar{E}$ at $25 \leq Z \leq 40$ is very similar. There is a limiting Reynolds number between $Re = 150$ and $Re = 200$ where the effect of the energy becoming more localised in the top and bottom of the pipe is equal to the effect of the increase in magnitude of E. For $e = 1$, the limiting Reynolds number is in a similar range. This does not occur for $e = 2$ before $Re = 240$ but the rate of increase of $\bar{E}$ decreases. This suggests that the limiting Reynolds number is in excess of $Re = 240$, and that changing e changes the limits of this behaviour.

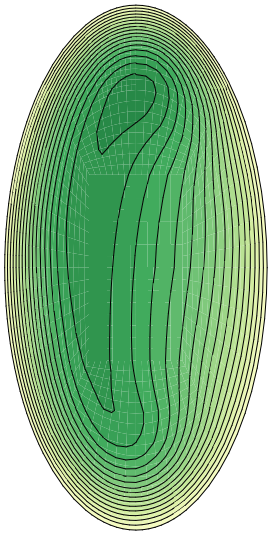


(a) U_{c7} $\gamma = 0.0080$ (b) $\gamma = 0.02$

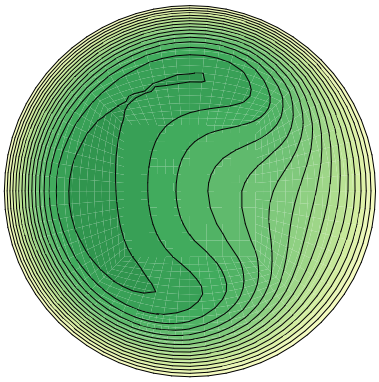
(c) U_{t3} $\gamma = 0.035$ (d) $\gamma = 0.08$

(e) U_{c2} $\gamma = 0.14$ (f) $\gamma = 0.15$

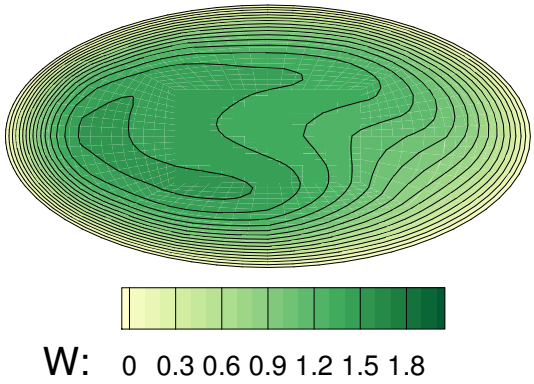
Figure 4.9: Figures (a), (c) and (e) show computed axial flow for $e = 1$ at $Re = 200$ for each pipe centreline at $Z = 35$. The outside wall is on the left of the page. Figures (b), (d) and (f) show axial flow for $Re = 1000$, $Dn = 100$, $\lambda = 0.01$ from Liu & Masliyah (1993). The outside wall for these plots is on the right of the page.



(a) $e = 0.5$

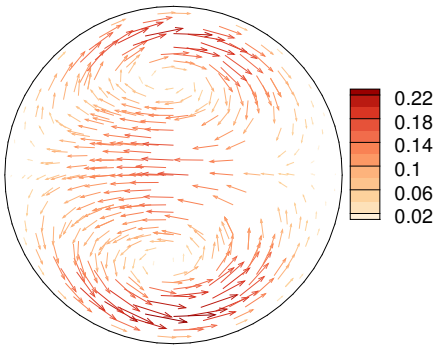


(b) $e = 1$

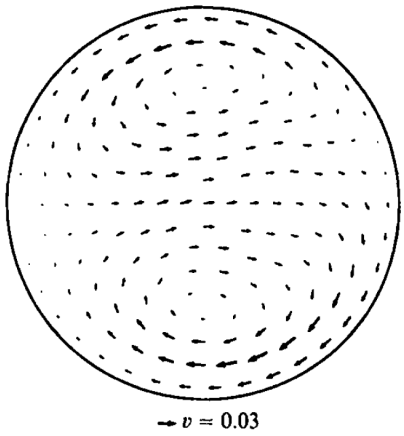


(c) $e = 2$

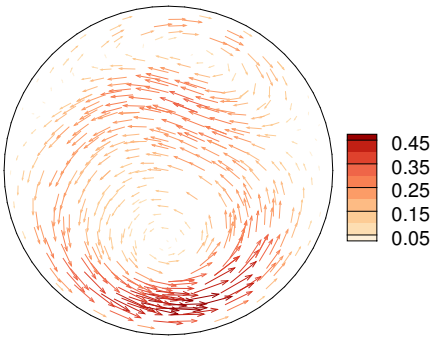
Figure 4.10: Fully developed, steady axial flow for pipe U_{t3} at $Re = 200$. The outside wall is on the left of the page.



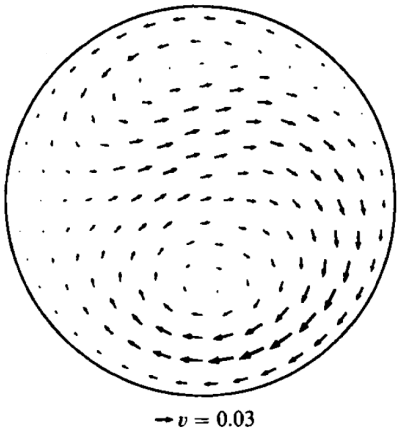
(a) U_{c7} $\gamma = 0.0080$



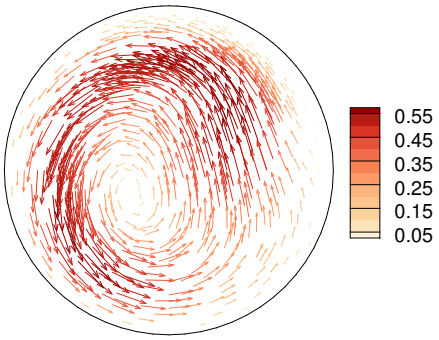
(b) $\gamma = 0.01$



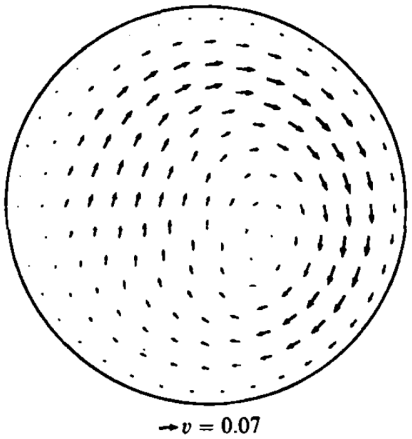
(c) U_{t3} $\gamma = 0.035$



(d) $\gamma = 0.04$

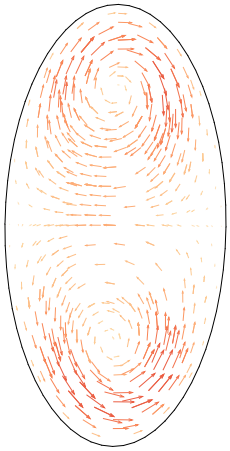


(e) U_{c2} $\gamma = 0.14$

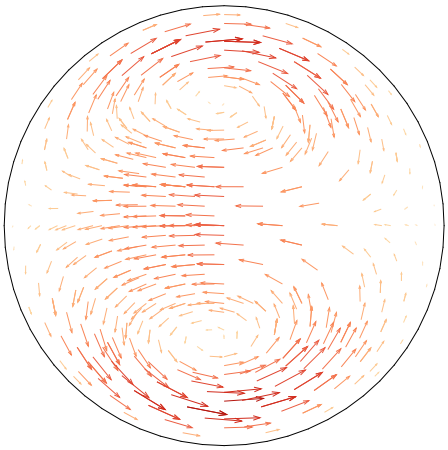


(f) $\gamma = 0.15$

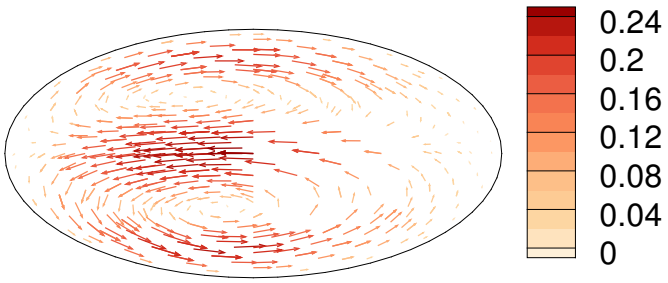
Figure 4.11: Comparison between computed vector field and results from (Liu & Masliyah 1993). Figures (a), (c) and (e) show the computed result at $Re = 200$ at $Z = 35$. The outside wall is on the left of the page. Figures (b), (d) and (f) show the results from (Liu & Masliyah 1993). The parameters for this flow are $Re = 1000$, $Dn = 100$ and $\lambda = 0.01$. The outside wall is on the right of the page.



(a) $e = 0.5$



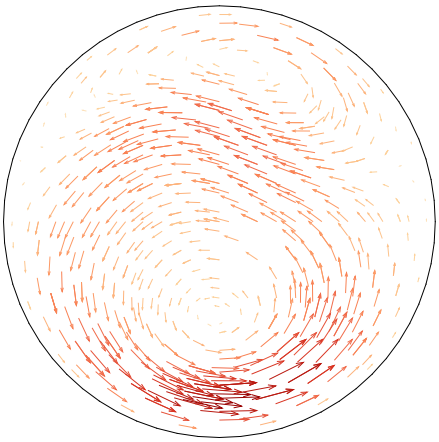
(b) $e = 1$



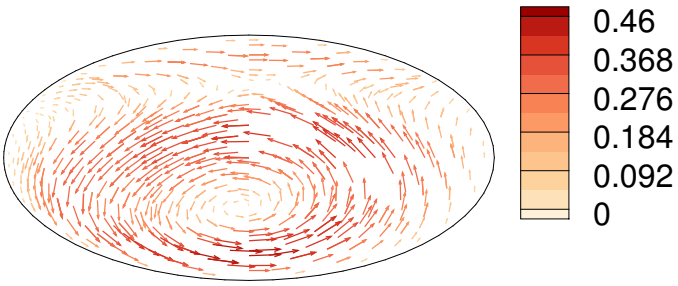
(c) $e = 2$

Figure 4.12: Fully developed cross-flow vector field at $Z = 35$ and $Re = 200$ for U_{c7}. The colour bar shows the magnitude of the vector $\sqrt{U^2 + V^2}$. The outside wall is on the left of the page.

(a) $e = 0.5$

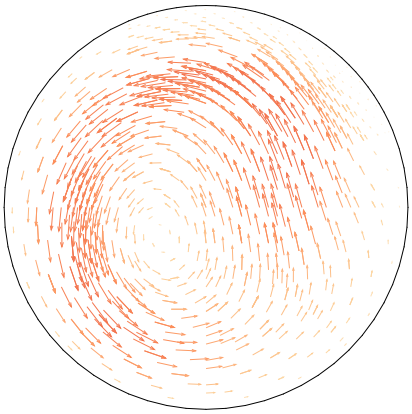


(b) $e = 1$

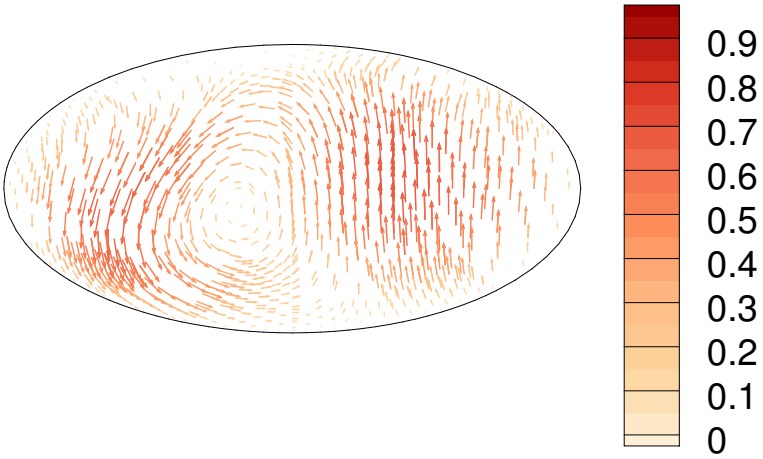


(c) $e = 2$

Figure 4.13: Fully developed cross-flow vector field at $Z = 35$ and $Re = 200$ for U_{t3}. The colour bar shows the magnitude of the vector $\sqrt{U^2 + V^2}$. The outside wall is on the left of the page.

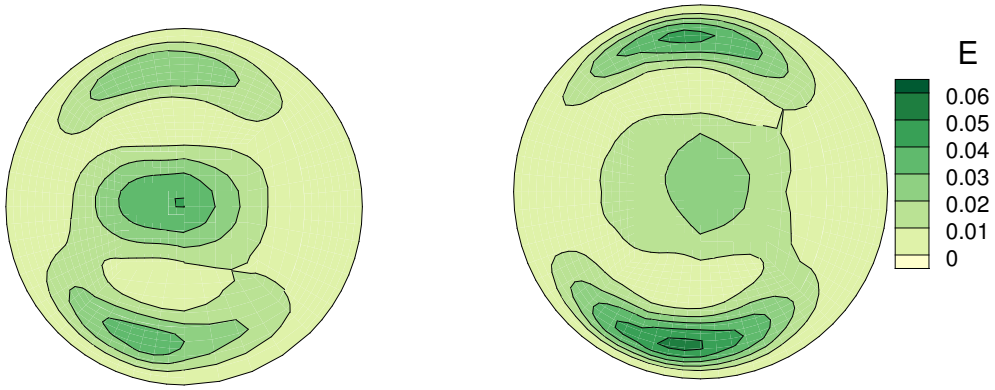


(a) $e = 1$



(b) $e = 2$

Figure 4.14: Fully developed cross-flow vector field at $Z = 35$ and $Re = 200$ for U_{c2}. The colour bar shows the magnitude of the vector $\sqrt{U^2 + V^2}$. The outside wall is on the left of the page.



(a) $Re = 100$ (b) $Re = 200$

Figure 4.15: Steady cross-flow energy at $Z = 35$ and $e = 1$ for U_{c7}. The outside wall is on the left of the page.

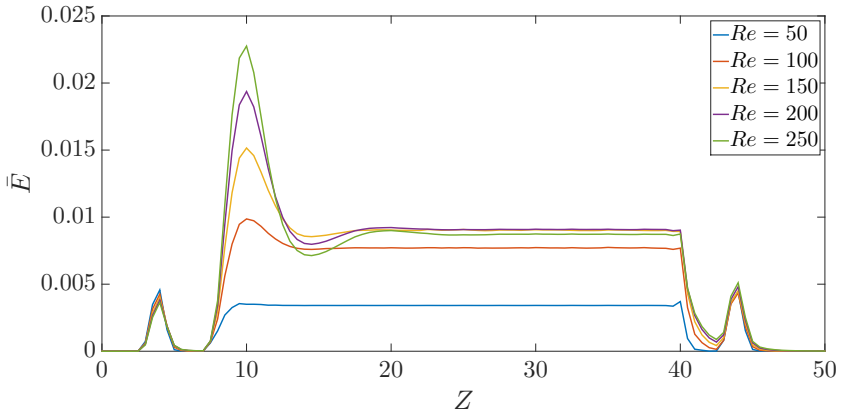


Figure 4.16: Cross-sectionally averaged energy along the length of the pipe for $e = 0.5$ for U_c7.

e/Re	140	150	160	170	180	180	190	200
0.5	X	O	7.82	7.82	7.82	7.82	7.82	7.82
0.62	X	X	X	O	8.51	8.51	8.51	8.51
0.7	X	X	X	X	X	X	O	10.72

Table 4.5: Wavelength and occurrence of oscillations periodic in length in the cross-flow energy for U_{c2}, $e = 0.5$. An X indicates the behaviour not occurring, an O indicates the onset of the periodic behaviour and a number indicates the period of the behaviour in terms of the centreline parameter ϕ, which represents the angular displacement of the spatial period. One turn of the helix corresponds to $\Delta\phi = 2\pi$.

4.1.3 Flows periodic in length

Unlike the other cases, the flow for U_{c2} when $e = 0.5$ does not become fully developed. Figure 4.6 shows that, for the $e = 0.5$ case, $\bar{E}$ is periodic in length. A periodic behaviour is also observed in the axial flow, cross-flow and pressure gradient. To further investigate this behaviour, additional simulations were performed. These simulations considered pipes with different values of e and used the long pipe length from table 4.3. Table 4.5 shows a range of aspect ratios and Reynolds numbers where the periodic behaviour of cross-sectional energy occurs. The period of the behaviour for a fixed centreline is dependent of e. The amplitude of the behaviour is dependent on Re. At higher Reynolds numbers, the average cross-sectional energy begins to show two modes of oscillation. This is shown in figure 4.17. There is also a noticeable change in the phase, as the development length of the periodic behaviour increases with Re.

The same flow is computed with an unsteady solver to determine the stability of the periodic solution. The same boundary conditions are used with the initial condition $(U, V, W) = \mathbf{0}$ everywhere in the pipe except the inlet. When the steady state is reached, the average cross-sectional energy is equal to the cross-sectional energy computed by the steady solver over the length of the pipe. The unsteady solver converges to the same solution as the steady solver with an initial condition that is very different from the steady solution. This shows that the solution found by the steady solver is stable.

Figures 4.18 and 4.19 show the axial and cross-flow over one period of oscillation. The length of one spatial period for $Re = 200$ in Z-space is 6.2. The location of the maximum magnitude of axial flow is near the upper inside wall for the period of oscillation, similar to the fully developed cases of $e = 1$ and $e = 2$. The shape of the axial contours deform slightly along the length of the period, but there is no fundamental change. The vector field has a

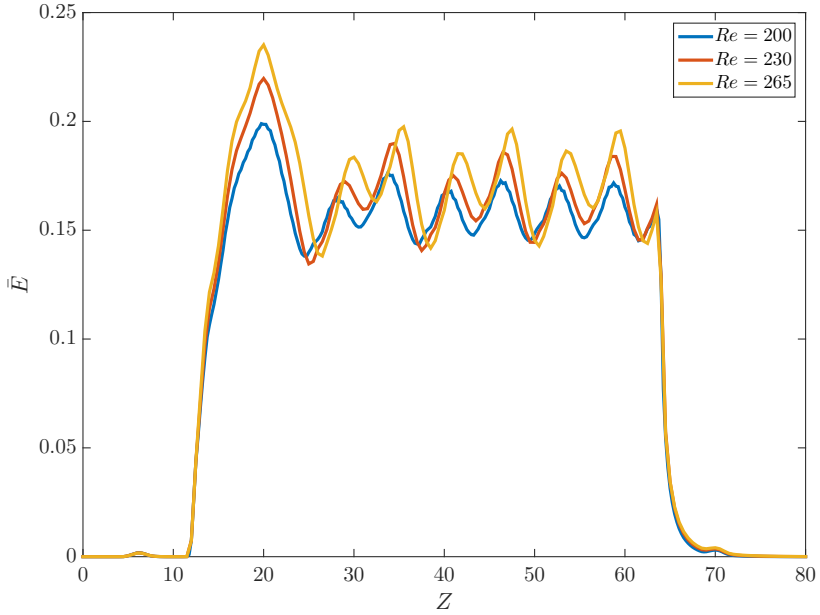
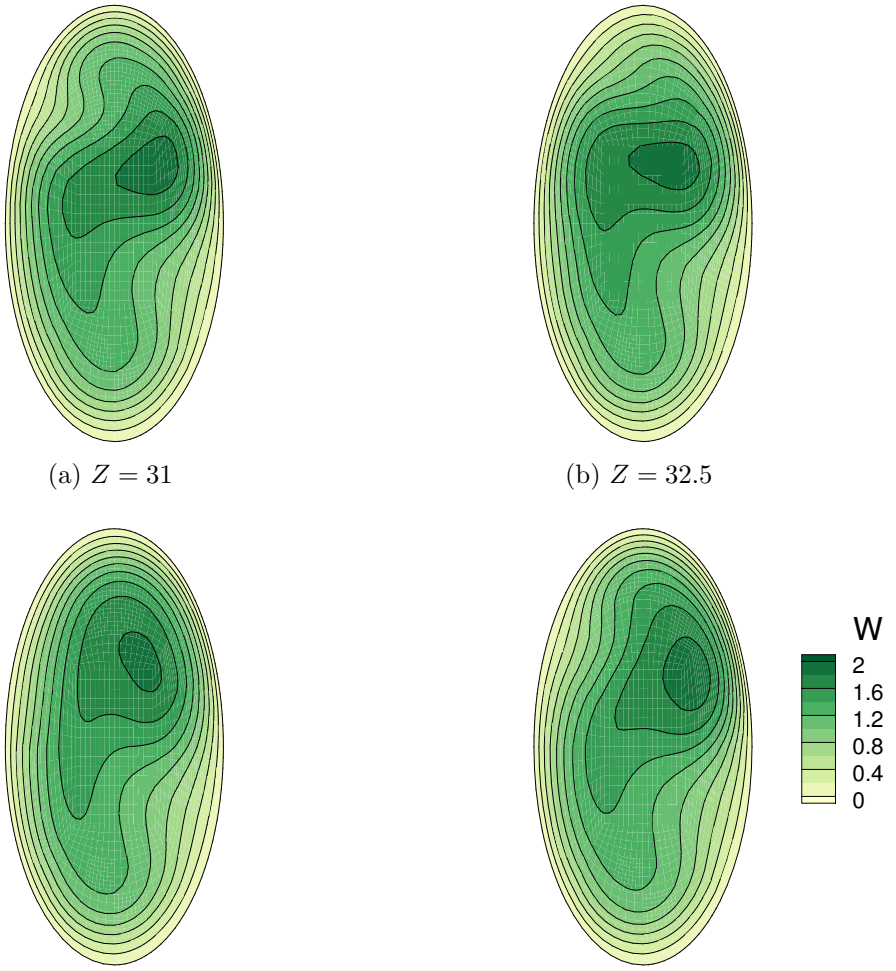


Figure 4.17: Cross-sectionally averaged cross-sectional energy along length of pipe U_{c2} at $e = 0.5$.

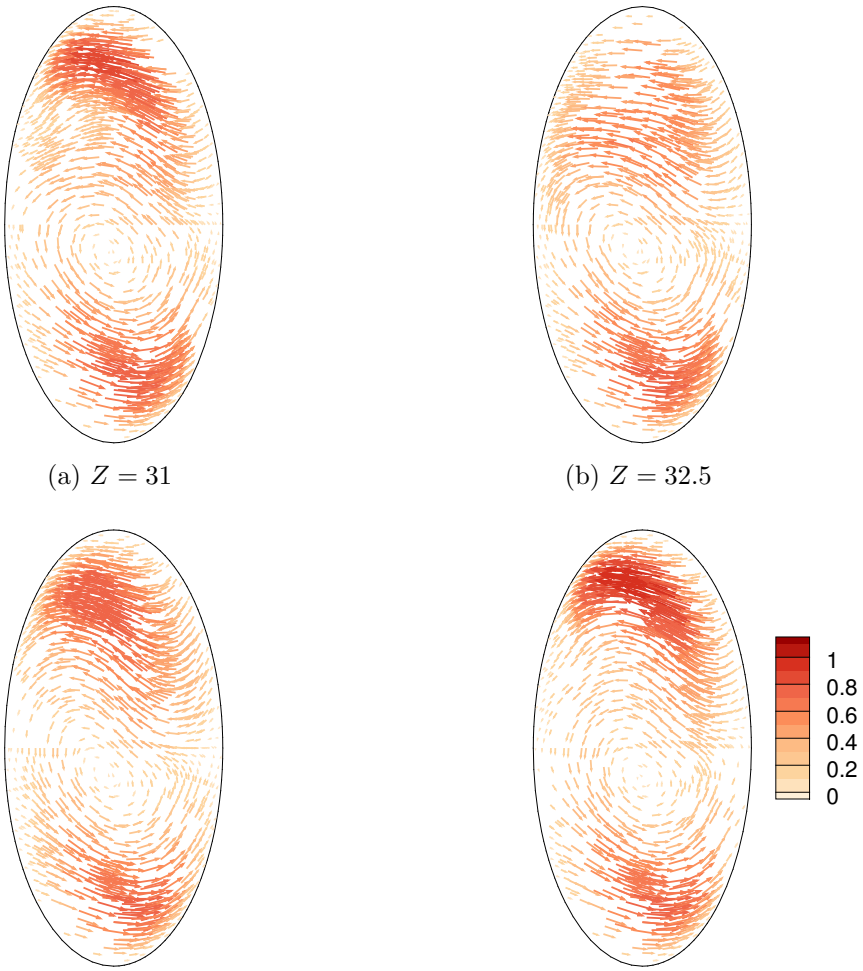
higher magnitude at $Z = 31$ and $Z = 35.5$, where $\bar{E}$ is highest. The general shape of the vector field is similar for each point along the wave. There is only one vortex present, with the flow at the bottom of the pipe moving to the inside wall and the flow at the top moving to the outside wall. The flow features do not fundamentally change over the period but the changes in the axial and cross-flow are still significant. There is more difference in these features at higher Reynolds numbers.



(a) $Z = 31$ (b) $Z = 32.5$

(c) $Z = 34$ (d) $Z = 35.5$

Figure 4.18: Axial flow for U_{c2} at $e = 0.5$ and $Re = 200$. The wavelength is $Z = 6.2$. The outside wall of the pipe is on the left of the page.



(a) $Z = 31$ (b) $Z = 32.5$

(c) $Z = 34$ (d) $Z = 35.5$

Figure 4.19: Cross-flow vector field for U_{c2} at $e = 0.5$ and $Re = 200$. The colour bar shows the vector magnitude $\sqrt{U^2 + V^2}$. The wavelength is $Z = 6.2$. The outside wall is on the left of the page.

.

	$e = 0.5$	$e = 1$	$e = 2$
U_{c7}	two even vortices	two even vortices	two even vortices closer to the outside wall
U_{t3}	small top vortex large bottom vortex	small top vortex large bottom vortex	one large vortex at the bottom of the pipe
U_{c2}	one large vortex near the bottom	one large vortex near to the outside wall	one large vortex near to the outside wall

Table 4.6: Summary of cross-flows for each model and $e = 0.5$, $e = 1$, and $e = 2$.

4.1.4 Summary

When umbilical cords are deformed and twisted, it is possible for the aspect ratio of the cross-section of the cord to change in a way similar to the plastic pipes discussed in chapter 3. Levy et al. (2015) shows that, under the strain of being coiled, the cross-section of the pipe changed from $e = 1$ to $e = 0.5$, so it is likely that umbilical cords will deform in a similar way. Table 4.6 summarises the flow pattern for the three models at three values of e, and highlights that, when $e = 2$ for pipe U_{t3}, the flow structure is significantly different than at $e = 1$ or $e = 0.5$. The differences are less pronounced at $e = 1$ and $e = 0.5$. As it is likely that umbilical cords deform in a way where $e < 1$, than it is likely that flow structures in deformed umbilical cords are similar to umbilical cords with circular cross-sections.

4.2 Unsteady results

In this section, pulsatile flow through the previous helical pipes is used to investigate flow through umbilical arteries. Unlike flow in umbilical veins, the flow in arteries is pulsatile due to the forcing of the fetal heart (Wilke 2016). The axial velocity, cross-sectional flow and pressure drop through the pipes is compared and contrasted to steady flows through the same pipes. The appropriateness of approximating unsteady flows with steady results, and hence, the effect of unsteadiness at low Strouhal numbers, will also be investigated. In order to model the unsteady flow, an inlet profile that approximates arterial umbilical flow is chosen. A truncated Fourier series expansion approximating a simple 'saw-tooth' profile is used to describe the centreline velocity at the inlet. Using the dimensional parameters shown in

the survey in table 4.1, $Re = 200$ and $St = 2/41$ are chosen as appropriate dimensionless parameters. The tolerances and number of elements used to compute each solution at $Re = 200$ are shown in table 4.7. As the torsion of the pipe increases, more elements are needed to resolve the flow due to the sharper spatial gradients of cross-flow velocities, similar to steady flows. More elements are also needed when the magnitude of the centreline axial velocity is greater.

In order to model the blood flow in an umbilical artery, a velocity waveform that approximates the pulsatility of a fetal heart is needed. In Wilke (2016), the maximum in-utero velocity is chosen to be the centreline velocity at the inlet, with the solution developed by Womersley (1955) providing the full cross-sectional profile.

The work presented here follows from Wilke (2016) and shows the development of the saw-tooth approximation used to define the unsteady flow profile used in this research.

First, the saw-tooth wave that defines the centreline velocity is approximated by the Fourier series of the odd periodic extension of,

$$W(0,t) - W_0 = \begin{cases} (W_{\text{peak}} - W_0)\frac{t}{t_1} & 0 < t \leq t_1 \\ (W_{\text{peak}} - W_0)\frac{(T/2-t)}{T/2-t_1} & t_1 < t \leq T/2, \end{cases} \tag{4.1}$$

where W_{peak} is the maximum centreline velocity over the period, t_1 is the time when the peak velocity is reached and W_0 is the time-averaged centreline velocity. The Fourier sine series is

$$W(0,t) - W_0 = \sum_{n=1}^{\infty} b_n \sin\left(\frac{2n\pi t}{T}\right), \tag{4.2}$$

where

$$b_n = \frac{4}{T}\int_0^{\frac{T}{2}} (W(0,t) - W_0)\sin\left(\frac{2n\pi t}{T}\right) \mathrm{d}t. \tag{4.3}$$

Without truncation, the centreline velocity is

$$W(0,t) - W_0 = \frac{W_{\text{peak}} T^2}{2\pi^2 t_1 (T/2 - t_1)} \sum_{n=1}^{\infty} \frac{1}{n^2} \sin\left(\frac{2n\pi t_1}{T}\right) \sin\left(\frac{2n\pi t}{T}\right). \tag{4.4}$$

The variation of the centreline velocity is also written by evaluating Womersley's solution from equation (1.4) at the centreline,

$$W(0,t) - W_{\max} = \text{Re}\left[\sum_{n=1}^{N} -\frac{iP_n}{\omega_n \rho} e^{i\omega_n t}\right] \tag{4.5}$$

where $W_{\max}$ is the maximum of the steady parabolic axial velocity profile. An expression for the complex pressure coefficients $P_n = C_n + iD_n$ is found by equating (4.4) and (4.5). The pressure coefficients can then be expressed in terms of the natural flow parameters and the complex numbers $A_n + iB_n = \frac{1}{J_0(i^{3/2}\alpha_n)} - 1$, with $A_n, B_n \in \mathcal{R}$. The complex pressure coefficients are

$$C_n = -\frac{A_n D_n}{B_n} \tag{4.6}$$

and

$$D_n = \frac{\rho\omega_n W_{\text{peak}} T^2}{2\pi^2 t_1 B_n n^2 (T/2 - t_1)(1 + A_n^2/B_n^2)} \sin\left(\frac{2n\pi t_1}{T}\right). \tag{4.7}$$

The full axial velocity profile is then given by the sum of the harmonics in equation (1.4).

The constants in the above expressions were chosen by Wilke (2016) to match data from literature. The non-dimensional peak and mean velocities are chosen to be $W_{\text{peak}} = 3.24$, $W_0 = W_{\max} = 2$ and the number of harmonics of the truncated series is chosen as $N = 4$. The timing of the peak velocity within the period is chosen as $t_1 = 0.15T$. The centreline axial velocity profile at the inlet over one period is shown in figure 4.20. The full axial profile at a range of times is shown in figure 4.21.

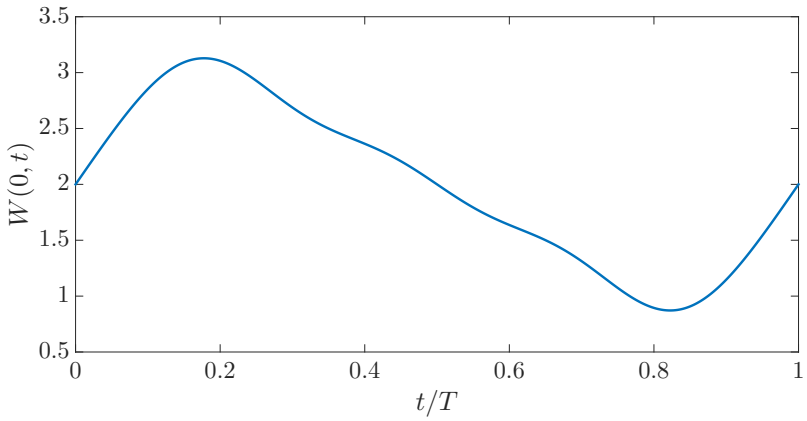
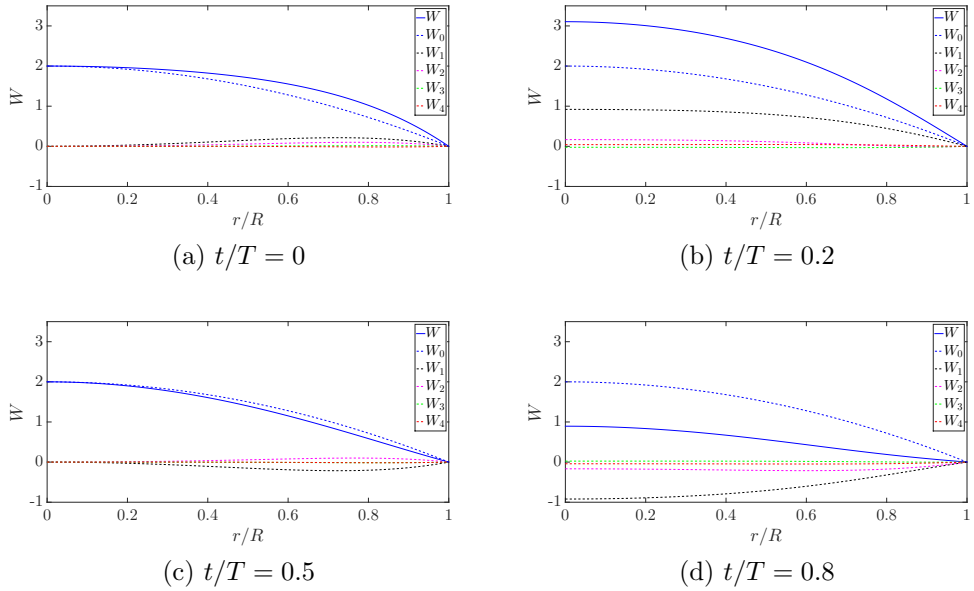


Figure 4.20: Saw-tooth inlet profile for the centreline velocity.

The velocity profile described above is used at the circular inlet of the pipe. The remaining boundary conditions are the same as those used in the steady simulations. There is a no-slip condition at the pipe walls and the outlet is psuedo-traction free. The initial condition is $(U, V, W) = \mathbf{0}$ everywhere in the pipe except for the inlet. At the beginning of the simulation,



(a) $t/T = 0$ (b) $t/T = 0.2$

(c) $t/T = 0.5$ (d) $t/T = 0.8$

Figure 4.21: Inlet profile for unsteady flows for $Re = 200$ and $St = 2/41$. The solid line shows the flow profile and the dashed lines show each harmonic.

the flow is initially at rest, and there is a sudden change at the inlet. This leads to very large temporal gradients.

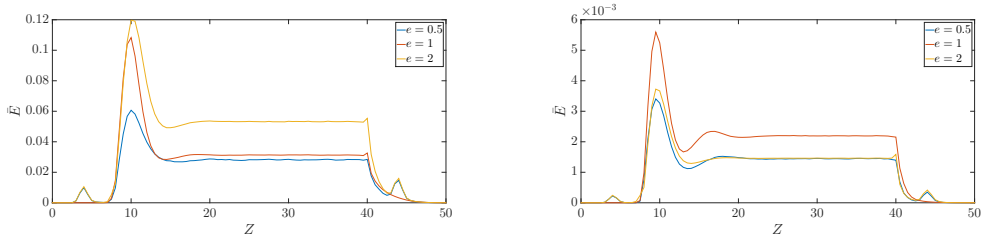
The pulsatile verification in section 2.7 has shown that a time step of $\Delta t = T/40$ is suitable to calculate pulsatile flow through a straight pipe at $Re = 200$. A time step of $\Delta t = T/40$ is chosen for all unsteady simulations except for pipe U_{c2} at $e = 1$. This simulation uses a time step of $\Delta t = T/50$ to ensure convergence of the non-linear Newton method used by `oomph-lib`. A time step of $\Delta t = 1/40$ led to non-convergence early in the solution where the temporal gradients are very large.

	e	MaxTol	MinTol	Elements at $t = 0.2$	Elements at $t = 0.8$
U_{c7}	$e = 0.5$	0.002	0.0002	6024	5807
	$e = 1$	0.0013	0.00013	8369	8551
	$e = 2$	0.0012	0.00012	5373	5051
U_{t3}	$e = 0.5$	0.0021	0.0021	12877	8222
	$e = 1$	0.0008	0.00008	16517	13479
	$e = 2$	0.002	0.0002	9503	8649
U_{c2}	$e = 0.5$	0.0015	0.00015	30748	16041
	$e = 1$	0.00101	0.000101	8621	8614
	$e = 2$	0.001	0.0001	20794	16006

Table 4.7: Tolerances and number of elements used for computations.

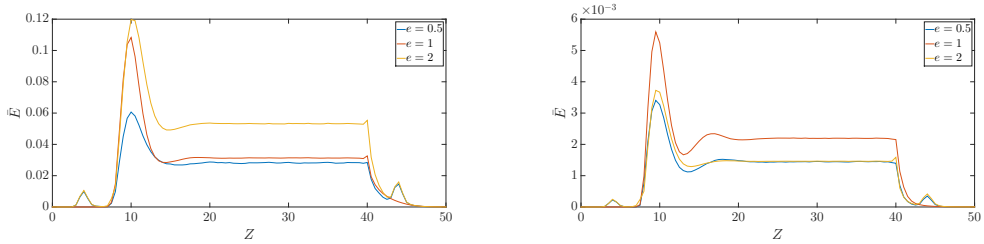
4.2.1 Spatial flow development

The unsteady flows develop spatially in a similar way to the steady flows, as shown in figures 4.22 to 4.24. There is a sharp increase in $\bar{E}$ when the pipe centreline transitions from straight to helical. Then, $\bar{E}$ settles to a fully developed state after a development length and then sharply decreases as the pipe changes from helical to straight. Most flows become fully developed at $Z \approx 20$ at both the maximum and minimum axial centreline velocity. There are two exceptions, the first is U_{c2} and $e = 1$ at $t/T = 0.8$, shown in figure 4.24b. This is at a time when the axial centreline velocity is near its minimum value. The flow has a much larger development length in this case, with the flow becoming fully developed at $Z \approx 30$. The U_{c2} flow at $e = 0.5$ does not become fully developed but, unlike the steady case, does not become periodic in length at any point in the period. Figure 4.25 shows the cross-sectionally averaged cross-flow energy for a range of times over the period for U_{c2} at $e = 0.5$. At no point in the period does the flow exhibit the magnitude of oscillation evident in the steady case.



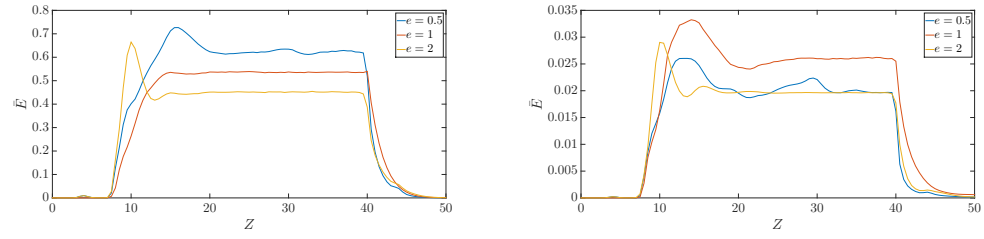
(a) $t/T = 0.2$ (b) $t/T = 0.8$

Figure 4.22: Cross-sectionally averaged energy along length of pipe U_{c7} at $Re = 200$ and $St = 2/41$.



(a) $t/T = 0.2$ (b) $t/T = 0.8$

Figure 4.23: Cross-sectionally averaged energy along length of pipe U_{t3} at $Re = 200$ and $St = 2/41$.



(a) $t/T = 0.2$ (b) $t/T = 0.8$

Figure 4.24: Cross-sectionally averaged energy along length of pipe U_{c2} at $Re = 200$ and $St = 2/41$.

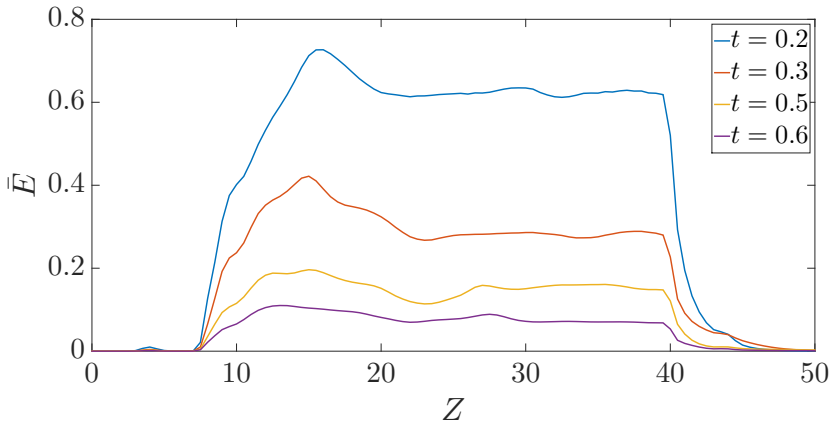


Figure 4.25: Cross-sectionally averaged energy for U_{c2} at $e = 0.5$, $Re = 200$ and $St = 2/41$ for a range of times in the period.

4.2.2 Fully developed flow

Figure 4.26 shows the axial flow for the unsteady flows at $t/T = 0.2$, which is approximately the time of maximum centreline axial velocity at the inlet, and $t/T = 0.8$, which is approximately the time of minimum centreline axial velocity, at $e = 1$. For flows through pipes U_{c7} and U_{t3}, the location of maximum axial velocity shifts to the outside wall of the pipe. This effect is more pronounced when the centreline velocity at the inlet is large. Figures 4.26f shows the location of maximum velocity near the inside wall for U_{c2} at $t/T = 0.2$, similar to the steady case. At $t/T = 0.8$, the location of maximum axial velocity is towards the top of the pipe.

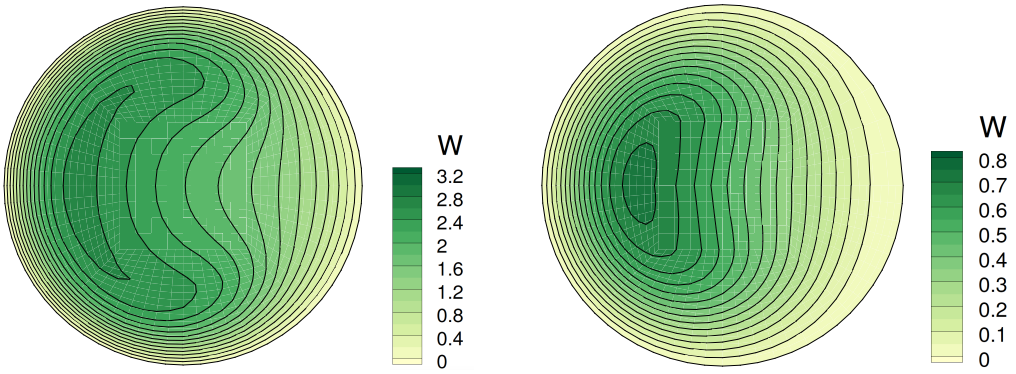
Figure 4.27 shows the effect of aspect ratio on the axial velocity for pipe U_{t3} at $t/T = 0.2$ for $Re = 200$ and $St = 2/41$. The general behaviour of the axial flow is similar with the steady case, shown in figure 4.10. The location of the maximum magnitude of axial velocity shifts to the outside wall and the contours deform to fit the new cross-sectional shape. The unsteady flows for the other pipes and aspect ratios are qualitatively similar to their steady counterparts.

The shape of the cross-flow for the unsteady case is qualitatively similar to the steady case. Table 4.6, which summarises steady cross-flows, also summarises the cross-flows for unsteady flows. Figures 4.28 to 4.36 compare the unsteady cross-flow vector fields for each centreline and aspect ratio at $t/T = 0.2$ and $t/T = 0.8$, for $Re = 200$ and $St = 2/41$ with the steady cross-flow at $Re = 200$. In each case, the magnitude of the cross-flow is more intense at $t/T = 0.2$ than at $t/T = 0.8$ or the steady case, as the axial velocity is larger. The steady and unsteady cross-flows are fundamentally similar in each case. The direction of the cross-flow and number of vortices are the same at $t/T = 0.2$, $t/T = 0.8$ and the steady case. The difference is vortex size is accentuated at $t/T = 0.2$ and is less prominent at $t/T = 0.8$ compared to the steady flow. For example, figure 4.32 shows that, at $t/T = 0.2$, the lower vortex is significantly larger than the upper vortex, but the difference is less pronounced in the steady case, and even less pronounced at $t/T = 0.8$.

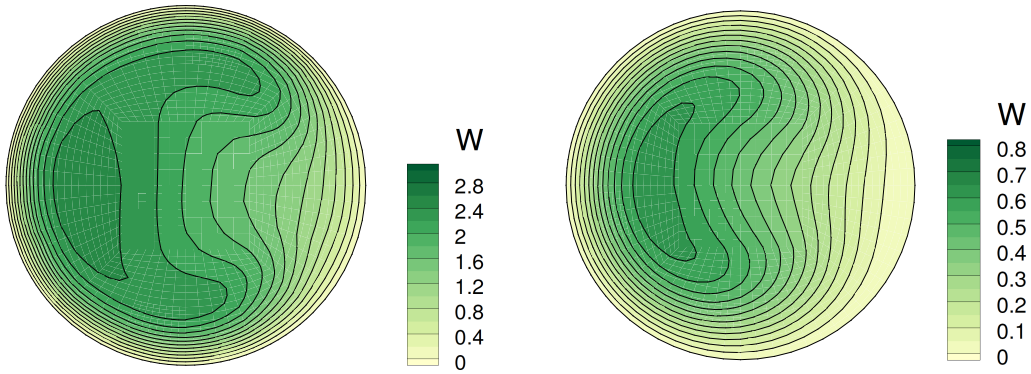
The cross-sectionally averaged cross-flow magnitude of the steady flow accurately predicts the cross-sectionally averaged cross-flow magnitude of the time-averaged unsteady flow. Table 4.8 compares the magnitudes of the fully developed cross-sectionally averaged cross-flow magnitude $\overline{\sqrt{E}} = \overline{\sqrt{U^2 + V^2}}$ for the steady case and the time-averaged and cross-sectionally averaged magnitude for the unsteady case $\overline{\sqrt{E}}_{\mathrm{TA}}$. The greatest percentage difference is 3.5%, which is for pipe U_{c7} at $e = 1$.

	e	$\overline{\sqrt{E}}$	$\overline{\sqrt{E}}_{\mathrm{TA}}$	% difference from steady
U_{c7}	0.5	0.0954	0.0958	0.42
	1	0.113	0.109	-3.5
	2	0.122	0.122	0
U_{t3}	0.5	0.200	0.200	0
	1	0.207	0.206	-0.483
	2	0.248	0.245	-1.21
U_{c2}	0.5	0.405	0.401	-0.988
	1	0.364	0.353	-3.02
	2	0.385	0.388	0.779

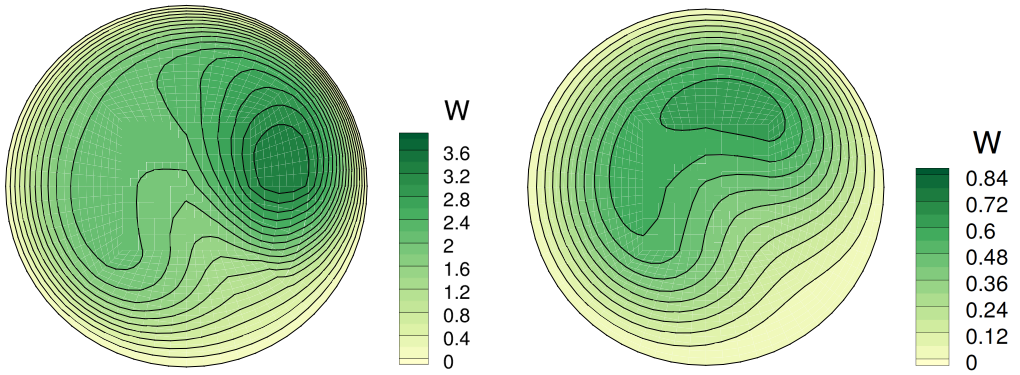
Table 4.8: Steady and time-averaged cross-sectionally averaged cross-flow magnitude, for $Re = 200$ and $St = 2/41$. The table highlights two results, the first being that at low and moderate torsion, $\overline{\sqrt{E}}$ generally increases as e increases. Normo-coiled umbilical cords are in this range of torsion. The second result is that, for these dynamic parameters, the change in average cross-flow energy is minimal, meaning steady flows can be used to approximate unsteady flows.



(a) U_{c7}, $t/T = 0.2$ (b) U_{c7}, $t/T = 0.8$

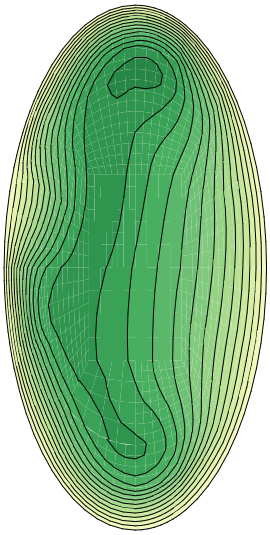


(c) U_{t3}, $t/T = 0.2$ (d) U_{t3}, $t/T = 0.8$

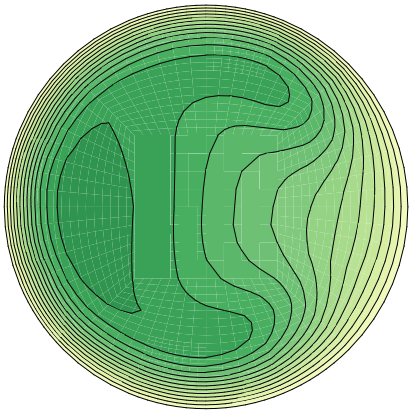


(e) U_{c2}, $t/T = 0.2$ (f) U_{c2}, $t/T = 0.8$

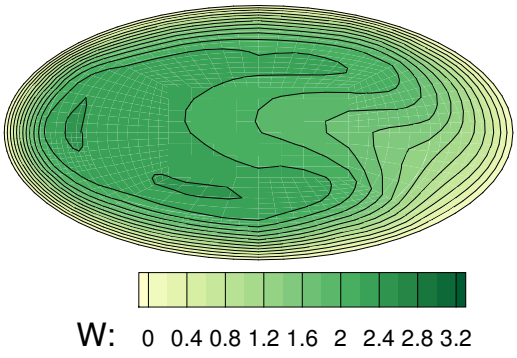
Figure 4.26: Axial Flow at $Z = 35$ for $e = 1$, $Re = 200$ and $St = 2/41$. The outside wall is on the left of the page.



(a) $e = 0.5$

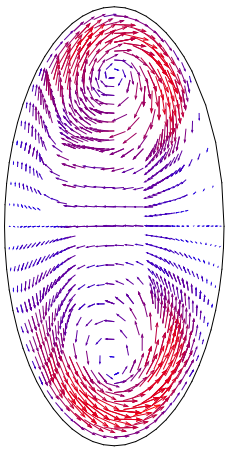


(b) $e = 1$

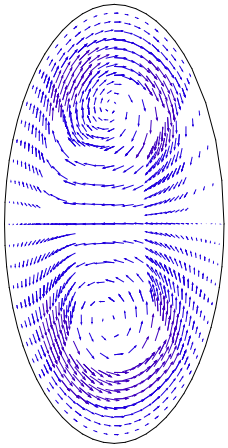


(c) $e = 2$

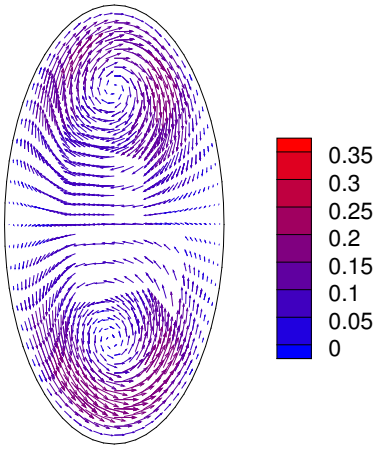
Figure 4.27: Fully developed axial flow for pipe U_{t3} at $t/T = 0.2$, $Z = 35$, $Re = 200$ and $St = 2/41$. The outside wall is on the left of the page.



(a) $t/T = 0.2$

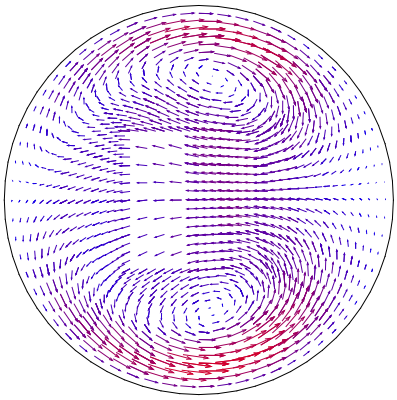


(b) $t/T = 0.8$

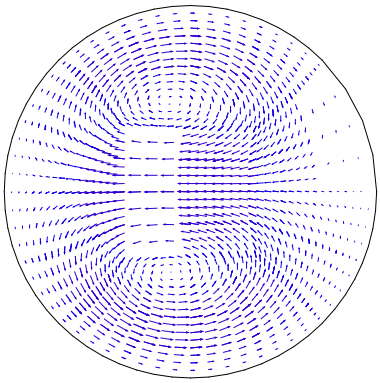


(c) steady

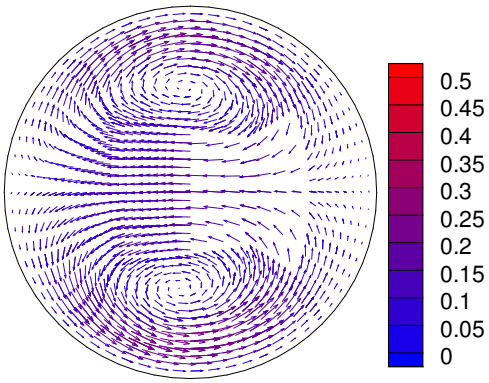
Figure 4.28: Comparison of unsteady cross-flow at peak and minimum centreline velocity and steady cross-flow for U_{c7} at $e = 0.5$, $Re = 200$ and $St = 2/14$. The colour bar shows the vector magnitude $\sqrt{U^2 + V^2}$ and the outside wall is on the left of the page.



(a) $t/T = 0.2$

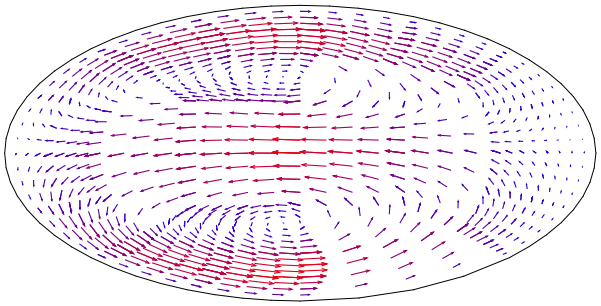


(b) $t/T = 0.8$

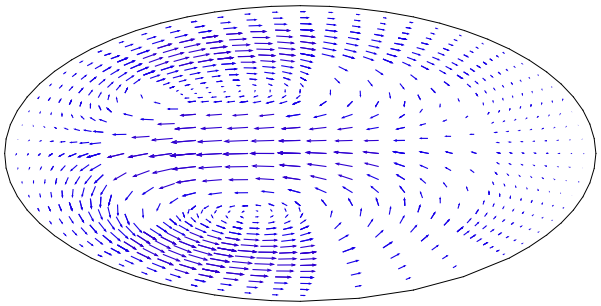


(c) steady

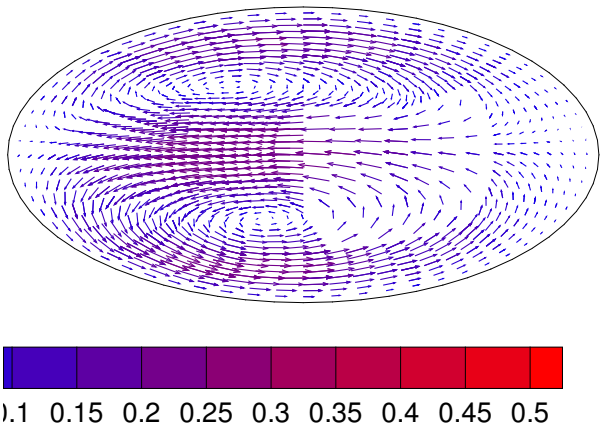
Figure 4.29: Comparison of unsteady cross-flow at peak and minimum centre-line velocity and steady cross-flow for U_{c7} at $e = 1$, $Re = 200$ and $St = 2/14$. The colour bar shows the vector magnitude $\sqrt{U^2 + V^2}$ and the outside wall is on the left of the page.



(a) $t/T = 0.2$

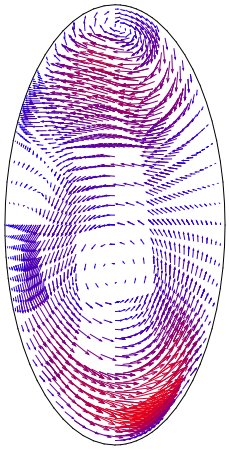


(b) $t/T = 0.8$

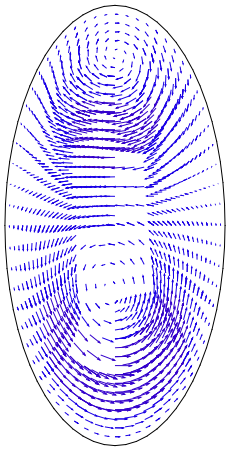


(c) steady

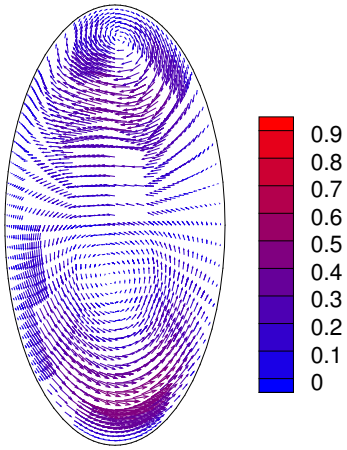
Figure 4.30: Comparison of unsteady cross-flow at peak and minimum centre-line velocity and steady cross-flow for U_{c7} at $e = 2$, $Re = 200$ and $St = 2/14$. The colour bar shows the vector magnitude $\sqrt{U^2 + V^2}$ and the outside wall is on the left of the page.



(a) $t/T = 0.2$

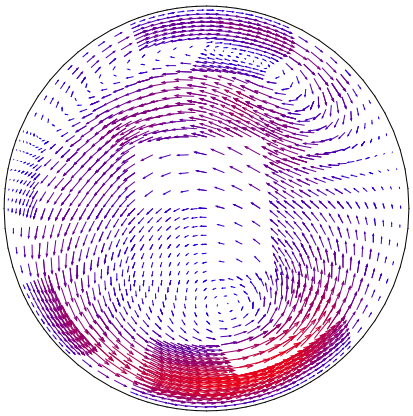


(b) $t/T = 0.8$

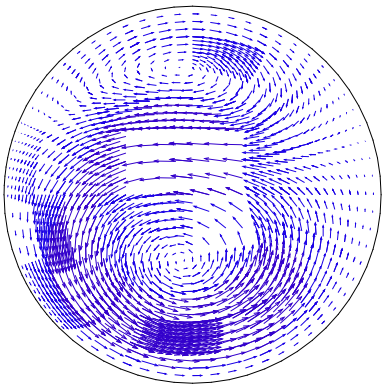


(c) steady

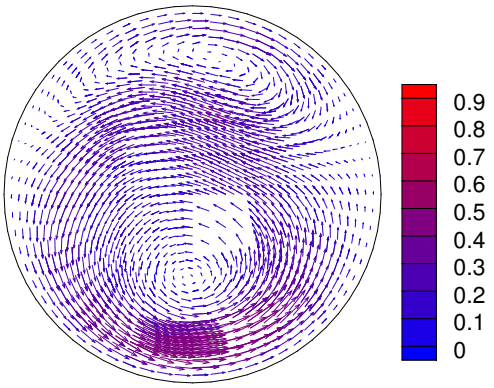
Figure 4.31: Comparison of unsteady cross-flow at peak and minimum centreline velocity and steady cross-flow for U_{t3} at $e = 0.5$, $Re = 200$ and $St = 2/14$. The colour bar shows the vector magnitude $\sqrt{U^2 + V^2}$ and the outside wall is on the left of the page.



(a) $t/T = 0.2$

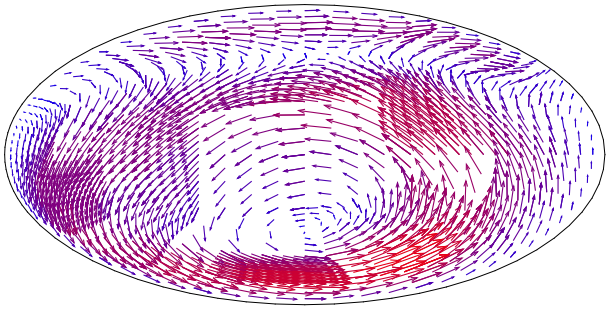


(b) $t/T = 0.8$

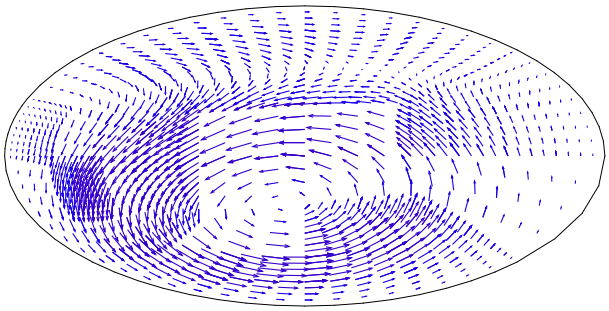


(c) steady

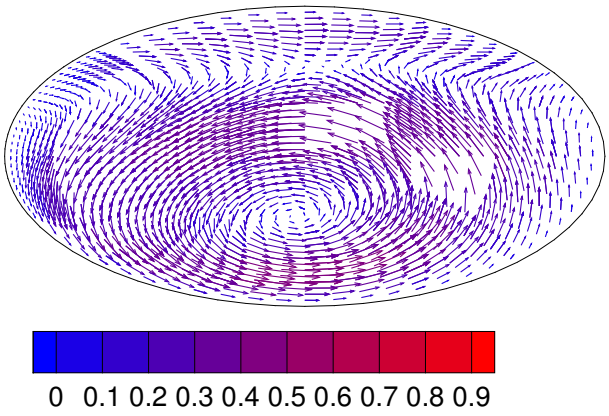
Figure 4.32: Comparison of unsteady cross-flow at peak and minimum centre-line velocity and steady cross-flow for U_{t3} at $e = 1$, $Re = 200$ and $St = 2/14$. The colour bar shows the vector magnitude $\sqrt{U^2 + V^2}$ and the outside wall is on the left of the page.



(a) $t/T = 0.2$

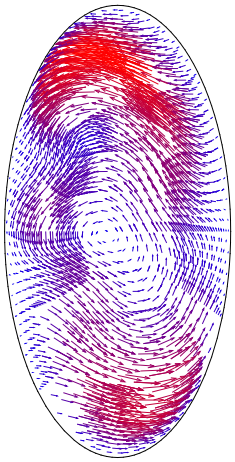


(b) $t/T = 0.8$

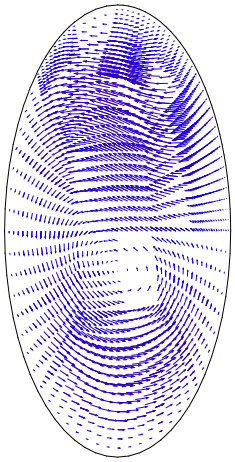


(c) steady

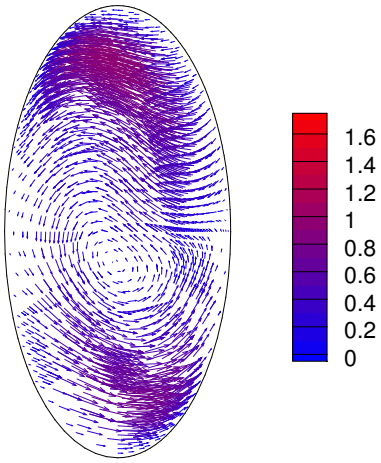
Figure 4.33: Comparison of unsteady cross-flow at peak and minimum centre-line velocity and steady cross-flow for U_{t3} at $e = 2$, $Re = 200$ and $St = 2/14$. The colour bar shows the vector magnitude $\sqrt{U^2 + V^2}$ and the outside wall is on the left of the page.



(a) $t/T = 0.2$

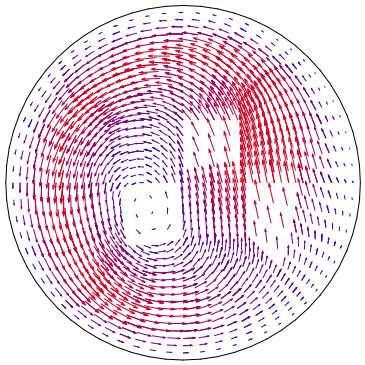


(b) $t/T = 0.8$

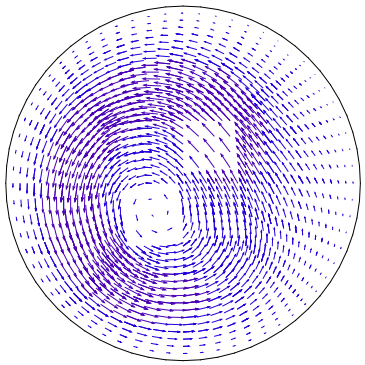


(c) steady

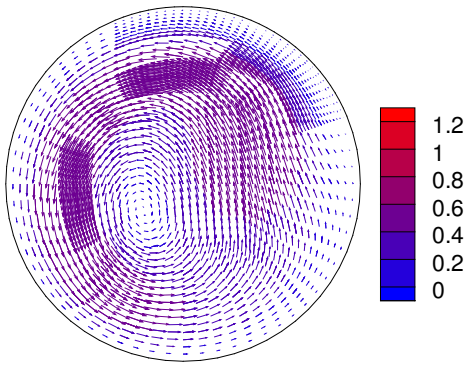
Figure 4.34: Comparison of unsteady cross-flow at peak and minimum centreline velocity and steady cross-flow for U_{c2} at $e = 0.5$, $Re = 200$ and $St = 2/14$. The colour bar shows the vector magnitude $\sqrt{U^2 + V^2}$ and the outside wall is on the left of the page.



(a) $t/T = 0.2$

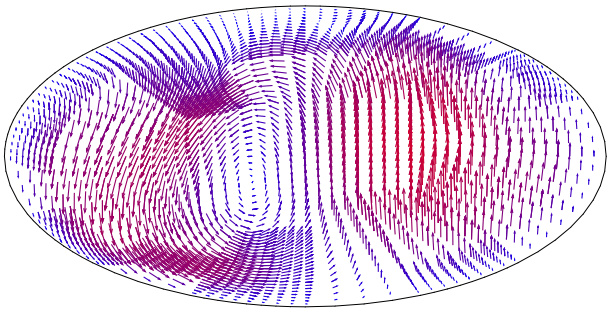


(b) $t/T = 0.8$

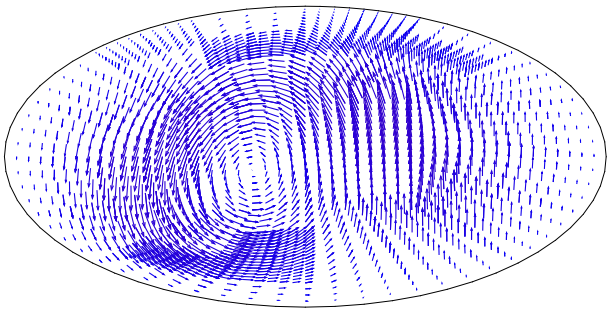


(c) steady

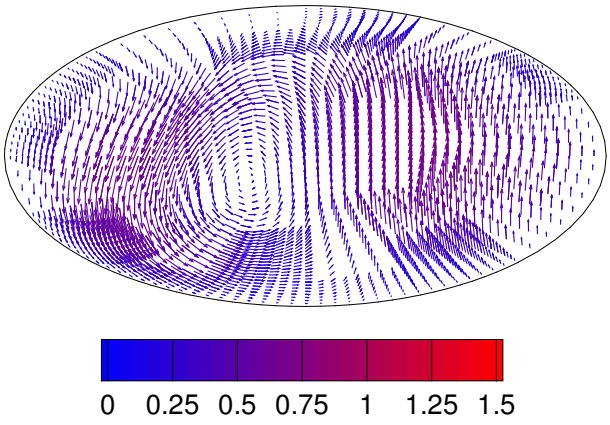
Figure 4.35: Comparison of unsteady cross-flow at peak and minimum centre-line velocity and steady cross-flow for U_{c2} at $e = 1$, $Re = 200$ and $St = 2/14$. The colour bar shows the vector magnitude $\sqrt{U^2 + V^2}$ and the outside wall is on the left of the page.



(a) $t/T = 0.2$



(b) $t/T = 0.8$



(c) steady

Figure 4.36: Comparison of unsteady cross-flow at peak and minimum centre-line velocity and steady cross-flow for U_{c2} at $e = 2$, $Re = 200$ and $St = 2/14$. The colour bar shows the vector magnitude $\sqrt{U^2 + V^2}$ and the outside wall is on the left of the page.

4.3 Pressure

It has been previously shown by Wilke (2016) that, for flows of small Strouhal number, the steady pressure gradient provides a good approximation for the time-averaged helical pressure gradient over one period, $(\partial P/\partial Z)_{\mathrm{TA}}$. The pressure gradient is of interest to flows through umbilical cords as it represents the loading of the fetal heart. The pressure in the pipe varies both over the cross-section and along the centreline. To simplify the calculation of $\partial P/\partial Z$, the cross-sectionally averaged pressure $\bar{P}$ is calculated at 101 points along the centreline and then numerically differentiated. The pressure gradient shown is then $\partial \bar{P}/\partial Z$. The mean value between $Z = 15$ and $Z = 37.5$ is then used to find $\partial \bar{P}/\partial Z$. The time-averaged pressure gradient is found by calculating

$$\left(\frac{\partial \bar{P}}{\partial Z}\right)_{\mathrm{TA}} = \frac{1}{T}\int_0^T \frac{\partial \bar{P}}{\partial Z}(t)\mathrm{d}t. \tag{4.8}$$

Using the relationship between the volumetric flow rate and the helical pressure gradient for steady laminar flow in a straight pipe given in Lekner (2007), the non-dimensional pressure gradient is found as a function of e. The relationship is

$$\frac{\partial P}{\partial Z} = -16\frac{e^2+1}{e}. \tag{4.9}$$

For $e = 1$, the circular case, this gives a pressure gradient of -32, and for both $e = 0.5$ and $e = 2$, this gives a pressure gradient of -40. For a straight pipe, the orientation of the ellipse does not affect the pressure gradient. Helical flow is more complex than straight pipe flow, and the presence of a cross-flow increases the pressure gradient for the same inlet condition. The orientation of the ellipse relative to the central helical axis also affects the pressure gradient.

Table 4.9 compares the steady and time-averaged helical pressure gradients for each case. The percentage difference between the $e \neq 1$ and $e = 1$ cases are shown for the steady case. The change in pressure gradient for U_{c2} from $e = 1$ to $e = 0.5$ is 30%. An increase of this magnitude in an umbilical cord could cause a loading on a fetal heart that is too large. The orientation of the ellipse affects the change of pressure by 19% for the U_{c2}. For pipes U_{c7} and U_{t3}, the pressure gradient is largest for $e = 2$. This contrasts with U_{c2}, where the highest pressure is found for $e = 0.5$. The orientation of the ellipse changes the pressure gradient and this change varies with the pipe centreline. The difference between $e = 1$ and $e \neq 1$ is significant, and the results for circular pipes do not accurately predict the results for elliptic pipes.

The steady pressure gradient is compared with the time-averaged unsteady pressure gradient for the same helical centreline and aspect ratio.

	e	$\frac{\partial \bar{P}}{\partial Z}$	% diff from $e = 1$	$\frac{\partial \bar{P}}{\partial Z}_{\mathrm{TA}}$	% diff from steady
U_{c7}	0.5	-49.0	9.9	-51.8	2.65
	1	-44.6	-	-47.4	3.14
	2	-53.0	19	-56.4	3.77
U_{t3}	0.5	-57.7	6.3	-60.0	3.99
	1	-54.3	-	-56.9	4.79
	2	-66.4	22	-69.2	4.22
U_{c2}	0.5	-68.9	30	-72.8	5.66
	1	-52.8	-	-54.5	3.22
	2	-62.9	11	-65.5	4.13

Table 4.9: Steady and time-averaged helical pressure gradient, for $Re = 200$ and $St = 2/41$. For all coils, $e \neq 1$ leads to a larger pressure gradient, which is similar to straight pipes. Time averaged pressure gradients are also generally 3% to 5% larger than their steady counterpart. Deformed umbilical cords increase the loading of the fetal heart in both the veins and arteries.

Wilke (2016) found that the time-average pressure drop could be estimated to within 5% of the steady case. This is shown to be the case for the helical pressure gradient for $e = 1$. For $e \neq 1$ the time-averaged unsteady pressure gradient can be estimated to within 6% of the steady pressure gradient. The aspect ratio has little effect on the relationship between steady and time-averaged unsteady pressure gradients.

Wilke (2016) developed a pressure index for quantifying umbilical flow resistance that accounts for loose knots and irregularly coiled cords. This index assumes that the umbilical cord has a circular cross-section everywhere. It is possible that the cross-section of the vessels, under constriction, may change into an ellipse, but it is very difficult to measure this in-utero. The change of cross-section has a significant effect on the pressure gradient, and the effect of this can be added to the pressure index in Wilke (2016) if the aspect ratio of the elliptic sections can be found.

4.4 Conclusion

Throughout this chapter, steady and unsteady flows through geometries representative of umbilical veins and arteries were presented. The aims were to investigate the effect of pipe centreline and aspect ratio on the axial flow, cross-flow and pressure gradient, and to investigate the effect of unsteadiness at low Strouhal numbers on the axial flow, cross-flow and pressure gradient. First, the effect of centreline and aspect ratio on the development of steady

flow was investigated. It was found that, in most cases, the flow becomes fully developed at $Z \approx 20$, which is a distance of $12.5D$ from the begging of the helical section. An exception to this was found for the U_{c2} centreline for values of $e < 1$. For sufficiently high Reynolds number and sufficiently low e, the flow becomes periodic in length. The non-dimensional parameter γ accurately predicts the behaviour of the fully developed axial and cross-flow for $e = 1$, but not for $e \neq 1$. For pipes with the same centreline, the behaviour of the fully developed axial flow through elliptic cross-sections is qualitatively similar at parameter values typical of flows through umbilical cords. The limits of γ that determine the behaviour of the cross-flow change with e.

Fully developed unsteady flow is qualitatively similar to fully developed steady flow at parameters typical of flow through umbilical arteries. This is shown for $e = 1$ in Wilke (2016), and is shown in this research for $e = 0.5$ and $e = 2$. The flow develops spatially in a similar way to steady flows. The fundamental features of the flow do not change when the flow is unsteady. The effect of e on the axial flow is minimal, and the effect on the cross-flow is similar to the effect of changing e for steady flows. The limits of γ that determine the number of vortices change with e. The time and cross-sectionally averaged cross-flow magnitude is accurately predicted by the steady results to within 3.5%.

The effect of aspect ratio and unsteadiness on the pressure gradient is investigated in this chapter. For a straight pipe with steady flow, changing the aspect ratio from $e = 1$ to $e \neq 1$ increases the pressure gradient. The orientation of the ellipse does not affect the pressure gradient for straight pipes but does affect the pressure gradient for helical pipes. The orientation of the ellipse relative to the central helical axis significantly affects the steady pressure gradient. For pipes U_{c7} and U_{t3}, which have low torsion, the pressure gradient increases the most for $e = 2$, where the larger axis is perpendicular to the helical centreline. For pipe U_{c2}, the pressure gradient is highest at $e = 0.5$, where the larger axis is parallel to the helical centreline. The pressure gradient for helical pipes with circular cross-sections does not accurately predict the pressure gradient for helical pipes with elliptic cross-sections. The time-averaged unsteady pressure gradients are accurately predicted by the steady pressure gradients for both circular and elliptic cross-sections. For the models investigated, the largest discrepancy between the steady and unsteady results is 5.7%. This is a significant result, as it allows the results from umbilical veins to be used to predict the results for umbilical arteries.

Chapter 5

Conclusions and future work

The aims of this study were

1. To develop and verify a `c++` code that uses that uses the `oomph-lib` library to find finite-element solutions of steady and unsteady flows through helical pipes of elliptic cross-sections.

2. To compare computational results with the experimentals performed in Levy et al. (2015).

3. To investigate the effect of the aspect ratio and orientation of the cross-section on steady and unsteady flow through typical models of umbilical cord vessels.

A computational model utilising the finite-element library `oomph-lib` was developed. Chapter 2 presents a verification of this method against the work of Germano (1989). Very good agreement between the theoretical work of Germano and the computational method were found at low Reynolds numbers and for small parameter ϵ, where the asymptotic solution presented in Germano (1989) is valid.

Chapter 3 compares the experimental work of Levy et al. (2015) with computed results. It was found that the cross-flow is very sensitive to the location of the plane that the results are viewed in. If the viewing plane is not normal to the pipe centreline, slight displacements of the viewing plane can drastically alter the resultant cross-flow field. This emphasises that great care must be taken when reporting experimental results. The axial velocity of suddenly stopping helical flow was found to be different to suddenly stopping flow in a straight pipe. The axial flow does not slow down in an axisymmetric manner. Instead, the location of maximum positive axial velocity remains towards the outside wall, and the reverse flow is stronger near the inside wall.

Chapter 4 investigates the effect of centreline parameters and aspect ratio on the development of flows with geometries representative of umbilical arteries and veins, and the fully developed state of these flows. A hypercoiled cord, a hypocoiled cord and a normocoiled cord were modelled. Remarkably, a behaviour periodic in length was found for the hypercoiled case at $e = 0.5$. This behaviour is not seen for any other pipe or in previous literature. Future work should consider the previously unseen periodic behaviour found in pipe U_{c2}. The limiting Reynolds number and aspect ratio were found for pipe U_{c2} but no other pipe centrelines were investigated. An effort should be made to find the dynamic and geometric limits of this remarkable behaviour. A range of changes to the structure of the flow were found to occur over the range of parameters of this research. Future work arising from this thesis should discuss the physical mechanisms that cause these changes to the flow structure.

The orientation of the ellipse of a helical pipe, unlike straight pipes, affects the pressure gradient of the flow. This is dependent on the pipe centreline. For low torsion, $e > 1$ increases the pressure gradient more than $e < 1$. This is not the case for pipes with large torsion, where the $e < 1$ has a larger effect on the pressure gradient.

Similar to the work of Wilke (2016), good agreement was found between steady results and time-averaged unsteady results. This was found to be case for both $e = 1$ and $e \neq 1$. This allows for results from umbilical veins to be used to accurately approximate time-averaged energies and pressures for umbilical arteries for cases where $e \neq 1$.

For physiological flows, like those in the umbilical cord, the interaction between the fluid and wall has a significant effect on the resultant flow (Weerappuli & Parakrama 1987). The software package `oomph-lib` has the ability to model these interactions. Future work should model these interactions and investigate how the fluid-structure interactions change the relationship between circular pipes and elliptical pipes.

Appendix A

Expression for cross-flow velocities in Germano (1989)

The expressions for the cross-flow velocities in flow through a helical pipe with an elliptic cross-section from (Germano 1989) are presented. Table A.1 describes the parameters used by Germano to define the cross-flow velocities and figure A.1 shows the coordinate system used.

Parameter	Meaning
Λ	a/b
λ	$\hat{\tau}/\hat{\kappa}$
ϵ	$\hat{\kappa}a$
U	centreline axial velocity for straight pipe under same pressure gradient
Re	Reynolds number Ua/ν
v_x	Component of velocity in $\mathbf{B}$ direction
v_y	Component of velocity in $-\mathbf{N}$ direction

Table A.1: Table of parameters used in Germano (1989).

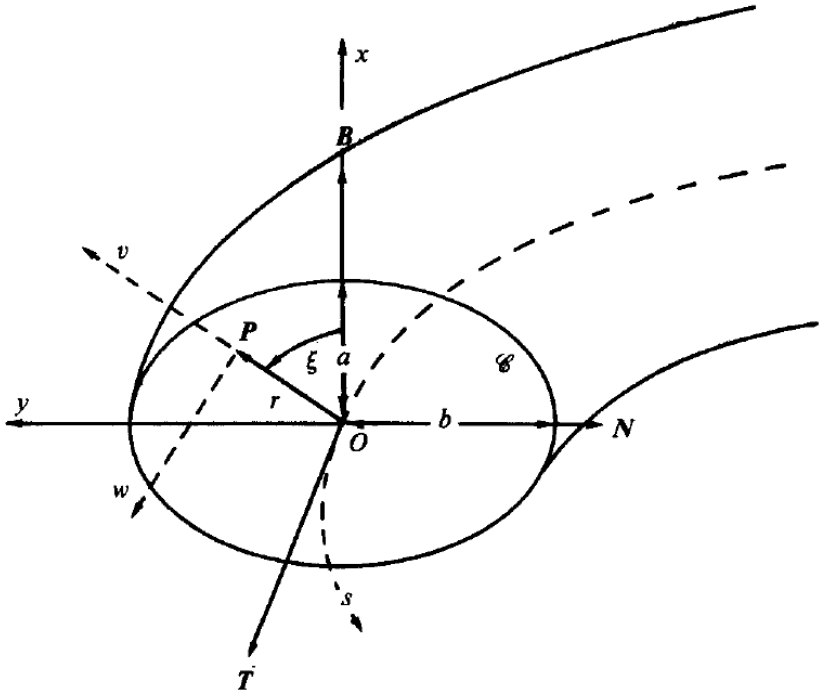


Figure A.1: Coordinate system used in Germano (1989) to find explicitly the Cartesian components of cross-flow motion.

The coefficients for the velocities are dependent on Λ, and are

$$F(\Lambda) = 35 + 84\Lambda^2 + 114\Lambda^4 + 20\Lambda^6 + 3\Lambda^8, \tag{A.1}$$

$$C_1 = \frac{375 + 820\Lambda^2 + 1114\Lambda^4 + 212\Lambda^6 + 39\Lambda^8}{180(5 + 2\Lambda^2 + \Lambda^4)F(\Lambda)}, \tag{A.2}$$

$$C_2 = -\frac{(75 + 2\Lambda^2 + 3\Lambda^4)}{180F(\Lambda)}, \tag{A.3}$$

$$C_3 = -\frac{(15 + 26\Lambda^2 + 39\Lambda^4)}{180F(\Lambda)} \tag{A.4}$$

and

$$h = \frac{1 + \Lambda^2}{6(1 + \Lambda^4) + 4\Lambda^2}. \tag{A.5}$$

The Cartesian components of the cross-flow are given explicitly by

$$v_x = 2\epsilon URe(1 - x^2 - \Lambda^2 y^2)$$
$$\left[-2C_2 x(1 - x^2 - \epsilon^2 y^2) + 4\Lambda^2 \left(h\frac{\lambda}{Re} + C_1 x + C_2 x y^2 + C_3 x^3\right) - \frac{\lambda}{2Re}\right] y, \tag{A.6}$$

and

$$v_y = 2\epsilon URe(1 - x^2 - \Lambda^2 y^2)$$
$$\left[(C_1 + C_2 y^2 + 3C_3 x^2)(1 - x^2 - \epsilon^2 y^2) - 4x\left(h\frac{\lambda}{Re} + C_1 x + C_2 x y^2 + C_3 x^3\right) + \frac{\lambda}{2Re}x\right] \tag{A.7}$$

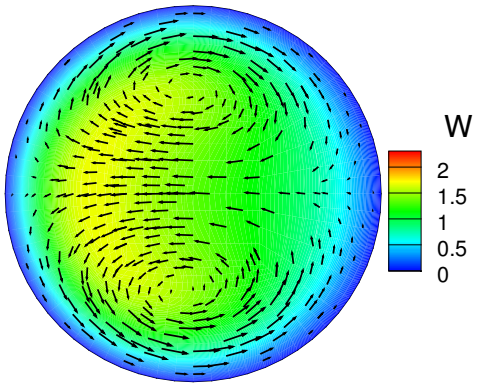
where $x = \hat{x}/a$ and $y = \hat{y}\Lambda/a$ are the non-dimensional displacements in the $\mathbf{B}$ and $-\mathbf{N}$ directions respectively.

Appendix B

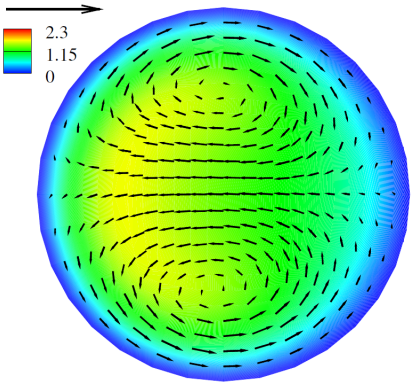
Comparison with Wilke (2016)

This appendix compares the work of this study and the work of Wilke (2016). Both works considered the same non-dimensional pipes with the same non-dimensional parameters and boundary conditions. The computed results in this appendix are the results from chapter 4, but presented in the same way as Wilke (2016). The outside wall of the pipe is on the left-hand side of the page.

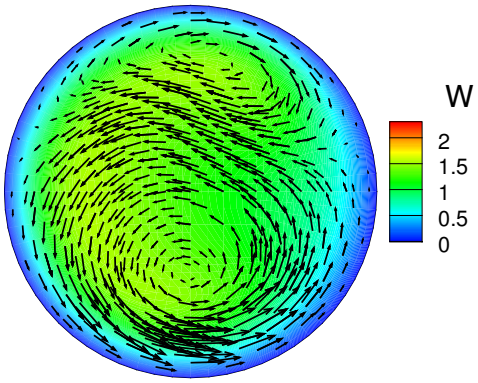
Figure B.1 compares the steady results from this work and the work of Wilke (2016). The results compare exceptionally well, despite transitioning from a straight section to a helical section differently. The pipe models in Wilke (2016) transition from straight to helical using a smooth curved section. The fully developed helical flow is independent of how the pipe transitions.



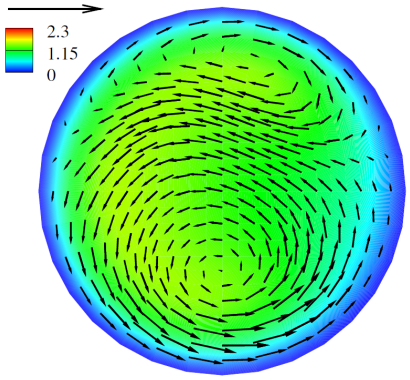
(a) computed



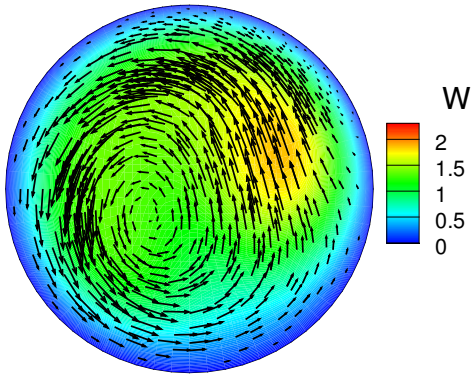
(b) Wilke (2016)



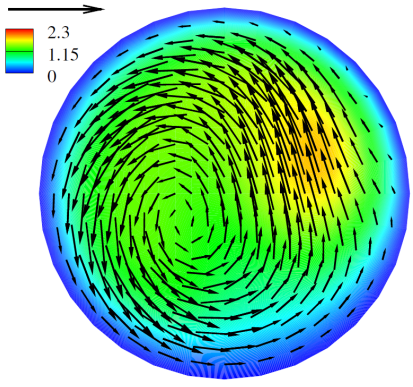
(c) computed



(d) Wilke (2016)

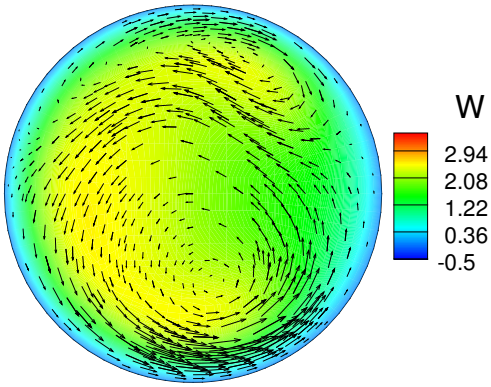


(e) computed

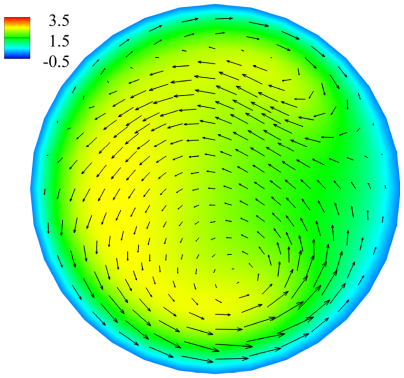


(f) Wilke (2016)

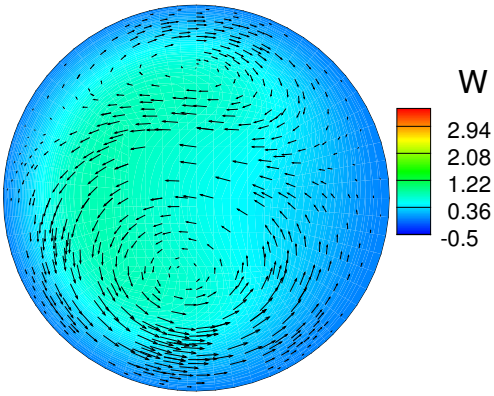
Figure B.1: Comparison between computed results and results from Wilke (2016) for the steady axial and cross-flow at $Re = 200$ and $e = 1$. The largest vector for U_{c7} in the computed results is 0.22. The largest vector for U_{t3} in the computed results is 0.45. The largest vector for U_{c2} in the computed results is 0.55.



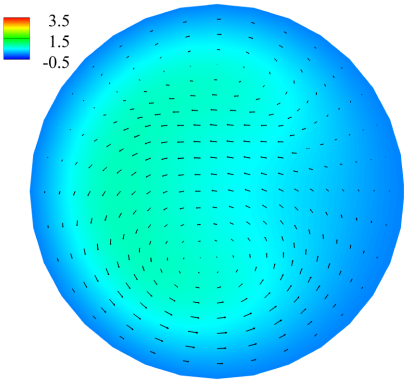
(a) computed $t = 0.15$



(b) Wilke (2016) $t = 0.15$



(c) computed $t = 0.75$



(d) Wilke (2016) $t = 0.76$

Figure B.2: Comparison between computed results and results from Wilke (2016) for the unsteady axial and cross-flow at for U_{t3} at $Re = 200$, $St = 2/41$, and $e = 1$.

Appendix C

Computer code

This appendix presents an example of the driver code for pulsatile flow through an umbilical artery.

```
//LIC// ====================================
//LIC// This file forms part of oomph-lib, the object-oriented,
//LIC// multi-physics finite-element library, available
//LIC// at http://www.oomph-lib.org.
//LIC//
//LIC//           Version 0.90. August 3, 2009.
//LIC//
//LIC// Copyright (C) 2006-2009 Matthias Heil and Andrew Hazel
//LIC//
//LIC// This library is free software; you can redistribute it
//LIC// and/or modify it under the terms of the
//LIC// GNU Lesser General Public License as published by the
//LIC// Free Software Foundation; either version 2.1 of the
//LIC// License, or (at your option) any later version.
//LIC//
//LIC// This library is distributed in the hope that it will be
//LIC// useful, but WITHOUT ANY WARRANTY;
//LIC// without even the implied warranty of
//LIC// MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
//LIC// See the GNU Lesser General Public License
//LIC// for more details.
//LIC//
//LIC// You should have received a copy of the GNU
//LIC// Lesser General Public License along with this library;
//LIC// if not, write to the Free Software Foundation, Inc.,
```

```
//LIC// 51 Franklin Street, Fifth Floor, Boston, MA
//LIC// 02110-1301  USA.
//LIC//
//LIC// The authors may be contacted at
//LIC// oomph-lib@maths.man.ac.uk.
//LIC//
//LIC//====================================================
///Driver for a 3D navier stokes steady entry flow problem
//in a helical tube

//Generic routines
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <complex.h>

#include "generic.h"
#include "navier_stokes.h"
#include <fstream>

// The mesh
#include "meshes/tube_mesh.h"

//Bessel Functions
#include "libAmosBessel.h"

using namespace std;

using namespace oomph;

//=start_of_namespace====================
/// Namespace for physical parameters
//=======================================
namespace Global_Physical_Variables
{

/// Reynolds number

//std::ifstream in_stream;

//Need to sort out order, fixed first then variables.
```

```
double Re = 0.0;
double St = 0.0;
double ReSt = 0.0;
double Pi = 4.0*atan(1.0);
//double Length = 1.0;
double Arc_Angle = 0.0;
double timer = 0.0;
double RadiusA,RadiusB,W,e,Delta,Pitch,Rc,tau,kappa,taubar,
lambda,kapbar,dt,t_stop,alpha;
double MinTol,MaxTol,PERIOD,a1,a2,g1,g11,g12,g2,
g21,g22,Restart_Time;
int checkRestart;
unsigned nsteps;
unsigned boundary_parameter;
char results_folder[50];
void get_variables();


} // end_of_namespace

//============================
//Get Variables
//============================

namespace Global_Physical_Variables
{
void get_variables()
{
std::ifstream in_stream;
in_stream.open("variables.dat");
in_stream >> RadiusA >> RadiusB >> W >> e >> Delta >> Rc >>
Pitch >> kappa >> tau >> taubar >> lambda >> kapbar >> Re >>
ReSt >> alpha >> PERIOD >> Arc_Angle >> nsteps >> dt >>
t_stop >> a1 >> a2 >> g1 >> g11 >> g12 >> g2 >> g21 >>
g22 >> MinTol >> MaxTol >> boundary_parameter >>
results_folder >> checkRestart >> Restart_Time;
in_stream.close();
}
}

namespace Mesh_Parameters {
```

```
double nlayer = 0;

void get_nlayer();
}

namespace Mesh_Parameters
{
void get_nlayer()
{
std::ifstream in_stream;
in_stream.open("nlayer.dat");
in_stream >> nlayer;
in_stream.close();

}
}

//==start_of_exact_solution=============
/// Namespace for exact solution
//==============================
namespace ExactSoln
{

/// Exact solution of the problem as a vector
void get_exact_u(const double& t, const Vector<double>& xe,
Vector<double>& u)
{
double x=xe[0];
double y=xe[1];
double z=xe[2];

double time = t;

const double pi = 4.0*atan(1.0);

//double RadA = Global_Physical_Variables::RadiusA;
//double RadB = Global_Physical_Variables::RadiusB;
double Kaps = Global_Physical_Variables::kappa;
double Taus = Global_Physical_Variables::tau;
```

```
double a1 = Global_Physical_Variables::a1;
double a2 = Global_Physical_Variables::a2;

double alpha_1 = Global_Physical_Variables::alpha;
double alpha_2 = Global_Physical_Variables::alpha*sqrt(2.0);
double u_mean = 1.0;

double PERIOD = Global_Physical_Variables::PERIOD;
double omega_1 = 2.0*pi*1.0/PERIOD;
double omega_2 = 2.0*pi*2.0/PERIOD;

cdouble beta_1 = 1.0/(sqrt(2.0))*(1.0fi-1.0)*alpha_1;
cdouble beta_2 = 1.0/(sqrt(2.0))*(1.0fi-1.0)*alpha_2;

cdouble BesselJ0_1;
cdouble BesselJ0_2;
cdouble BesselJ1_1;
cdouble BesselJ1_2;
cdouble BesselJ0_1r;
cdouble BesselJ0_2r;

AmosBessel('J', beta_1, 0.0, 1, 0, &BesselJ0_1);
AmosBessel('J', beta_1, 1.0, 1, 0, &BesselJ1_1);
AmosBessel('J', beta_2, 0.0, 1, 0, &BesselJ0_2);
AmosBessel('J', beta_2, 1.0, 1, 0, &BesselJ1_2);

cdouble complex_1 = 2.0*BesselJ1_1/(beta_1*BesselJ0_1) - 1.0;
cdouble complex_2 = 2.0*BesselJ1_2/(beta_2*BesselJ0_2) - 1.0;

double abscomplex_1 = creal(complex_1)*creal(complex_1)+
cimag(complex_1)*cimag(complex_1);
double abscomplex_2 = creal(complex_2)*creal(complex_2)+
cimag(complex_2)*cimag(complex_2);

double C_1 = -a1*omega_1*u_mean*creal(complex_1)/(abscomplex_1);

double D_1 = a1*omega_1*u_mean*cimag(complex_1)/(abscomplex_1);

double C_2 = -a2*omega_2*u_mean*cimag(complex_2)/(abscomplex_2);

double D_2 = -a2*omega_2*u_mean*creal(complex_2)/(abscomplex_2);
```

```
cdouble P_1 = C_1 + 1.0fi*D_1;
cdouble P_2 = C_2 + 1.0fi*D_2;

cdouble expbit_1 = cos(omega_1*time)+1.0fi*sin(omega_1*time);

cdouble expbit_2 = cos(omega_2*time)+1.0fi*sin(omega_2*time);

// (Re-)assign velocity profile at inflow values
//-----------------------------------------

double yandz = sqrt(pow(y,2)+pow(z,2));

double r_R = sqrt((pow(x,2))/(0.25)+(pow(yandz,2))/(0.25));
//double r_R = sqrt(pow(x,2)/(pow(RadA,2))
+pow(y,2)/(pow(RadB,2)));

AmosBessel('J', beta_1*r_R, 0.0, 1, 0, &BesselJ0_1r);
AmosBessel('J', beta_2*r_R, 0.0, 1, 0, &BesselJ0_2r);

// n = 0 is just poiselle flow

// flow for x direction

double u_0_y = 2.0*Kaps*(1.0-r_R*r_R);

double u_0_z = 2.0*Taus*(1.0-r_R*r_R);
//double u_0_z = 2.0*(1.0-r_R*r_R);

//n = 1 mode

double u_1_y = Kaps*(creal((P_1*1.0fi)/(omega_1)*
((BesselJ0_1r)/(BesselJ0_1)-1.0)*expbit_1));

double u_1_z = Taus*(creal((P_1*1.0fi)/(omega_1)*
((BesselJ0_1r)/(BesselJ0_1)-1.0)*expbit_1));
//double u_1_z = (creal((P_1*1.0fi)/(omega_1)*
((BesselJ0_1r)/(BesselJ0_1)-1.0)*expbit_1));

//n = 2 mode
```

```
double u_2_y = Kaps*(creal((P_2*1.0fi)/(omega_2)*
((BesselJ0_2r)/(BesselJ0_2)-1.0)*expbit_2));

double u_2_z = Taus*(creal((P_2*1.0fi)/(omega_2)*
((BesselJ0_2r)/(BesselJ0_2)-1.0)*expbit_2));
//double u_2_z = (creal((P_2*1.0fi)/(omega_2)*
((BesselJ0_2r)/(BesselJ0_2)-1.0)*expbit_2));

//Set Velocity Components
u.resize(3);

u[1]=u_0_y+u_1_y+u_2_y;

u[0]=0.0;

u[2]=u_0_z+u_1_z+u_2_z;

//double u_axial = sqrt(u[1]*u[1]+u[2]*u[2]);

//cout << "The solution at time " << time << " and at radius " <<
r_R << " for U is " << u_axial << std::endl;


}


/// Exact solution of the problem as a vector
void get_sawtooth_u(const double& t, const Vector<double>& xe,
Vector<double>& u)
{
double x=xe[0];
double y=xe[1];
double z=xe[2];

double time = t;

const double pi = 4.0*atan(1.0);

double Rho = 1.0;
double umax_t = 1.24;
double t1 = 0.15*Global_Physical_Variables::PERIOD;
```

```
//double RadA = Global_Physical_Variables::RadiusA;
//double RadB = Global_Physical_Variables::RadiusB;
double Kaps = Global_Physical_Variables::kappa;
double Taus = Global_Physical_Variables::tau;

double a1 = Global_Physical_Variables::a1;
double a2 = Global_Physical_Variables::a2;

double alpha_1 = Global_Physical_Variables::alpha;
double alpha_2 = Global_Physical_Variables::alpha*sqrt(2.0);
double alpha_3 = Global_Physical_Variables::alpha*sqrt(3.0);
double alpha_4 = Global_Physical_Variables::alpha*sqrt(4.0);
double u_mean = 1.0;

double T = Global_Physical_Variables::PERIOD;
double omega_1 = 2.0*pi*1.0/T;
double omega_2 = 2.0*pi*2.0/T;
double omega_3 = 2.0*pi*3.0/T;
double omega_4 = 2.0*pi*4.0/T;

cdouble beta_1 = 1.0/(sqrt(2.0))*(1.0fi-1.0)*alpha_1;
cdouble beta_2 = 1.0/(sqrt(2.0))*(1.0fi-1.0)*alpha_2;
cdouble beta_3 = 1.0/(sqrt(2.0))*(1.0fi-1.0)*alpha_3;
cdouble beta_4 = 1.0/(sqrt(2.0))*(1.0fi-1.0)*alpha_4;

cdouble BesselJ0_1;
cdouble BesselJ0_2;
cdouble BesselJ0_3;
cdouble BesselJ0_4;

cdouble BesselJ0_1r;
cdouble BesselJ0_2r;
cdouble BesselJ0_3r;
cdouble BesselJ0_4r;

AmosBessel('J', beta_1, 0.0, 1, 0, &BesselJ0_1);
AmosBessel('J', beta_2, 0.0, 1, 0, &BesselJ0_2);
AmosBessel('J', beta_3, 0.0, 1, 0, &BesselJ0_3);
AmosBessel('J', beta_4, 0.0, 1, 0, &BesselJ0_4);
```

```
cdouble complex_1 = 1.0/(BesselJ0_1) - 1.0;
cdouble complex_2 = 1.0/(BesselJ0_2) - 1.0;
cdouble complex_3 = 1.0/(BesselJ0_3) - 1.0;
cdouble complex_4 = 1.0/(BesselJ0_4) - 1.0;

double A_1 = creal(complex_1);
double B_1 = cimag(complex_1);
double A_2 = creal(complex_2);
double B_2 = cimag(complex_2);
double A_3 = creal(complex_3);
double B_3 = cimag(complex_3);
double A_4 = creal(complex_4);
double B_4 = cimag(complex_4);

double D_1 = 1.0/(1.0 + (A_1*A_1)/(B_1*B_1))*
(Rho*omega_1)/(B_1*1.0*1.0)*
(umax_t*T*T)/(2.0*pi*pi*t1*(T/2.0 - t1))*sin((2.0*pi*1.0*t1)/T);
double C_1 = (-A_1*D_1)/B_1;

double D_2 = 1.0/(1.0 + (A_2*A_2)/(B_2*B_2))*
(Rho*omega_2)/(B_2*2.0*2.0)*
(umax_t*T*T)/(2.0*pi*pi*t1*(T/2.0 - t1))*sin((2.0*pi*2.0*t1)/T);
double C_2 = (-A_2*D_2)/B_2;

double D_3 = 1.0/(1.0 + (A_3*A_3)/(B_3*B_3))*
(Rho*omega_3)/(B_3*3.0*3.0)*
(umax_t*T*T)/(2.0*pi*pi*t1*(T/2.0 - t1))*sin((2.0*pi*3.0*t1)/T);
double C_3 = (-A_3*D_3)/B_3;

double D_4 = 1.0/(1.0 + (A_4*A_4)/(B_4*B_4))*
(Rho*omega_4)/(B_4*4.0*4.0)*
(umax_t*T*T)/(2.0*pi*pi*t1*(T/2.0 - t1))*sin((2.0*pi*4.0*t1)/T);
double C_4 = (-A_4*D_4)/B_4;

cdouble P_1 = C_1 + 1.0fi*D_1;
cdouble P_2 = C_2 + 1.0fi*D_2;
cdouble P_3 = C_3 + 1.0fi*D_3;
cdouble P_4 = C_4 + 1.0fi*D_4;

cdouble expbit_1 = cos(omega_1*time)+1.0fi*sin(omega_1*time);
```

```
cdouble expbit_2 = cos(omega_2*time)+1.0fi*sin(omega_2*time);

cdouble expbit_3 = cos(omega_3*time)+1.0fi*sin(omega_3*time);

cdouble expbit_4 = cos(omega_4*time)+1.0fi*sin(omega_4*time);
// (Re-)assign velocity profile at inflow values
//-----------------------------------------

double yandz2 = y*y+z*z;

double r_R = sqrt((pow(x,2))/(0.25)
+(yandz2)/(0.25));
//double r_R = sqrt(pow(x,2)/(pow(RadA,2))+
pow(y,2)/(pow(RadB,2)));

AmosBessel('J', beta_1*r_R, 0.0, 1, 0, &BesselJ0_1r);
AmosBessel('J', beta_2*r_R, 0.0, 1, 0, &BesselJ0_2r);
AmosBessel('J', beta_3*r_R, 0.0, 1, 0, &BesselJ0_3r);
AmosBessel('J', beta_4*r_R, 0.0, 1, 0, &BesselJ0_4r);

// n = 0 is just poiselle flow

// flow for x direction

double u_0_y = 2.0*Kaps*(1.0-r_R*r_R);

double u_0_z = 2.0*Taus*(1.0-r_R*r_R);
//double u_0_z = 2.0*(1.0-r_R*r_R);

//n = 1 mode

double u_1_y = Kaps*(creal((P_1*1.0fi)/(omega_1)*
((BesselJ0_1r)/(BesselJ0_1)-1.0)*expbit_1));

double u_1_z = Taus*(creal((P_1*1.0fi)/(omega_1)
*((BesselJ0_1r)/(BesselJ0_1)-1.0)*expbit_1));


//n = 2 mode

double u_2_y = Kaps*(creal((P_2*1.0fi)/(omega_2)*
```

```
((BesselJ0_2r)/(BesselJ0_2)-1.0)*expbit_2));

double u_2_z = Taus*(creal((P_2*1.0fi)/(omega_2)*
((BesselJ0_2r)/(BesselJ0_2)-1.0)*expbit_2));

//n = 3 mode

double u_3_y = Kaps*(creal((P_3*1.0fi)/(omega_3)*
((BesselJ0_3r)/(BesselJ0_3)-1.0)*expbit_3));

double u_3_z = Taus*(creal((P_3*1.0fi)/(omega_3)*
((BesselJ0_3r)/(BesselJ0_3)-1.0)*expbit_3));

// n = 4 mode

double u_4_y = Kaps*(creal((P_4*1.0fi)/(omega_2)*
((BesselJ0_4r)/(BesselJ0_4)-1.0)*expbit_4));

double u_4_z = Taus*(creal((P_4*1.0fi)/(omega_2)*
((BesselJ0_4r)/(BesselJ0_4)-1.0)*expbit_4));

//Set Velocity Components
u.resize(3);

u[1]=u_0_y+u_1_y+u_2_y+u_3_y+u_4_y;

u[0]=0.0;

u[2]=u_0_z+u_1_z+u_2_z+u_3_z+u_4_z;

//double u_axial = sqrt(u[1]*u[1]+u[2]*u[2]);

//cout << "The solution at time " << time << " and at radius " <<
r_R << " for U is " << u_axial << std::endl;


    }


/// Exact solution of the problem as a vector for Rc=0
//USE THIS WHEN STRAIGHT PIPE ALONG Z AXIS
```

```
void get_exact_u_Rc(const double& t, const Vector<double>& xe,
Vector<double>& u)
{
double x=xe[0];
double y=xe[1];
double z=xe[2];

double time = t;

const double pi = 4.0*atan(1.0);

//double RadA = Global_Physical_Variables::RadiusA;
// double RadB = Global_Physical_Variables::RadiusB;

double a1 = Global_Physical_Variables::a1;
double a2 = Global_Physical_Variables::a2;

double alpha_1 = Global_Physical_Variables::alpha;
double alpha_2 = Global_Physical_Variables::alpha*sqrt(2.0);
double u_mean = 1.0;

double PERIOD = Global_Physical_Variables::PERIOD;
double omega_1 = 2.0*pi*1.0/PERIOD;
double omega_2 = 2.0*pi*2.0/PERIOD;

cdouble beta_1 = 1.0/(sqrt(2.0))*(1.0fi-1.0)*alpha_1;
cdouble beta_2 = 1.0/(sqrt(2.0))*(1.0fi-1.0)*alpha_2;

cdouble BesselJ0_1;
cdouble BesselJ0_2;
cdouble BesselJ1_1;
cdouble BesselJ1_2;
cdouble BesselJ0_1r;
cdouble BesselJ0_2r;

AmosBessel('J', beta_1, 0.0, 1, 0, &BesselJ0_1);
AmosBessel('J', beta_1, 1.0, 1, 0, &BesselJ1_1);
AmosBessel('J', beta_2, 0.0, 1, 0, &BesselJ0_2);
AmosBessel('J', beta_2, 1.0, 1, 0, &BesselJ1_2);

cdouble complex_1 = 2.0*BesselJ1_1/(beta_1*BesselJ0_1) - 1.0;
```

```
cdouble complex_2 = 2.0*BesselJ1_2/(beta_2*BesselJ0_2) - 1.0;

double abscomplex_1 = creal(complex_1)*creal(complex_1)+
cimag(complex_1)*cimag(complex_1);
double abscomplex_2 = creal(complex_2)*creal(complex_2)+
cimag(complex_2)*cimag(complex_2);

double C_1 = -a1*omega_1*u_mean*
creal(complex_1)/(abscomplex_1);

double D_1 = a1*omega_1*u_mean*
cimag(complex_1)/(abscomplex_1);

double C_2 = -a2*omega_2*u_mean*
cimag(complex_2)/(abscomplex_2);

double D_2 = -a2*omega_2*u_mean*
creal(complex_2)/(abscomplex_2);

cdouble P_1 = C_1 + 1.0fi*D_1;
cdouble P_2 = C_2 + 1.0fi*D_2;

cdouble expbit_1 = cos(omega_1*time)+1.0fi*sin(omega_1*time);

cdouble expbit_2 = cos(omega_2*time)+1.0fi*sin(omega_2*time);

// (Re-)assign velocity profile at inflow values
//-----------------------------------------

double r_R = sqrt((pow(x,2))/(0.25)+(pow(y,2))/(0.25));

AmosBessel('J', beta_1*r_R, 0.0, 1, 0, &BesselJ0_1r);
AmosBessel('J', beta_2*r_R, 0.0, 1, 0, &BesselJ0_2r);

// n = 0 is just poiselle flow

// flow for x direction

double u_0_z = 2.0*(1.0-r_R*r_R);

//n = 1 mode
```

```
double u_1_z = (creal((P_1*1.0fi)/(omega_1)
*((BesselJ0_1r)/(BesselJ0_1)-1.0)*expbit_1));

//n = 2 mode

double u_2_z = (creal((P_2*1.0fi)/(omega_2)*
((BesselJ0_2r)/(BesselJ0_2)-1.0)*expbit_2));

//Set Velocity Components
u.resize(3);

u[1]=0.0;

u[0]=0.0;

u[2]=u_0_z+u_1_z+u_2_z;

//double u_axial = sqrt(u[1]*u[1]+u[2]*u[2]);

//cout << "The solution at time " << time << " and at radius " <<
r_R << " for U is " << u_axial << std::endl;


}

void get_square(const double& t, double& f)
{

unsigned n_limit = 75;

double c = 75.0;

double F_term;

double f_inc = 0.5;

double filter;

double pi = 4.0*atan(1.0);
```

```
for(unsigned k=0; k<n_limit; k++)
{   F_term = (2.0/pi) * 1.0/(2.0*k + 1.0) *
sin((2.0*k + 1.0)*pi*t);

filter = exp((-(2.0*k + 1.0)*(2.0*k + 1.0))/(2.0*c*c));

f_inc = f_inc + F_term*filter;
}

f = f_inc;
}


} // end of exact_solution

namespace oomph
{


//======customised_TH element==================
/// Customised TH element -- simply overloads the output function.
/// All other functionality is retained.
/// This ouputs the 'local' tube coordinates of a regular cord
//=====================================
//template<unsigned DIM>
    class CustomisedRefineableQTaylorHoodElement :
public virtual RefineableQTaylorHoodElement<3>
    {

    public:

/// Empty constructor
CustomisedRefineableQTaylorHoodElement(){};

/// Empty virtual constructor
~CustomisedRefineableQTaylorHoodElement(){};

/// Overload output function
void output(std::ostream& outfile,const unsigned &nplot)
{
```

```
Vector<double> s(3);
// Vector<double> xe(3);
// Vector<double> soln(3);

//double Pi = 4.0*atan(1.0);
//BEGIN ARCLENGTH HACK

//Require pi
//double Pi = 4.0*atan(1.0);
double nlayer=Mesh_Parameters::nlayer;

//Storage for number of layers in intial mesh
// - adjust as appropriate
// (See call to constructor of RefineableTubeMesh below)
//unsigned nlayer=Mesh_Parameters::nlayer;

//Storage for centreline limits of initial mesh
// - adjust as appropriate
// (See call to constructor of RefineableTubeMesh below)
//double centreline_start=0.;
//double centreline_end=nlayer;

//Calculate the layer of the initial mesh to which
//the macro element belongs
unsigned macro_number=this->macro_elem_pt()->
macro_element_number();
//finds the element of the initial mesh to which the
//current element belongs.
unsigned layer=unsigned(macro_number/5);
//number of layers before 'current' layer.
integer division gives this value.

//Calculate the start coordinates and thickness of that layer
double g_start_macro_el=layer;
double layer_thickness=1.0;

//Storage for position within macro element
Vector<double> s_macro(3); //store position within macro element

//END ARCLENGTH HACK
```

```
// Tecplot header info
outfile << this->tecplot_zone_string(nplot);

//loop over all the points
unsigned num_plot_points=this->nplot_points(nplot);
for (unsigned iplot=0;iplot<num_plot_points;iplot++)
// for every plot point
{
//BEGIN ARCLENGTH HACK

this->get_s_plot(iplot,nplot,s);

//Obtain the position within the macro element of our
//current local coordinate s
for(unsigned i=0;i<3;i++)
{
s_macro[i]=this->s_macro_ll(i)+0.5*(s[i]+1.0)*
(this->s_macro_ur(i)-this->s_macro_ll(i));
}

//Compute zeta from second coordinate of position within macro element
//together with macro element's location in mesh as calculated above
double g=g_start_macro_el+0.5*(s_macro[2]+1.0)*layer_thickness;

//END ARCLENGTH HACK


double x = interpolated_x(s,0);
double y = interpolated_x(s,1);
double z = interpolated_x(s,2);

// xe[0] = x;
// xe[1] = y;
// xe[2] = z;

//  double time = Global_Physical_Variables::timer;
//if (z==0){
//cout << "The solution at time " << time << endl;}
//  ExactSoln::get_exact_u(time,xe,soln);

// Velocities
```

```
// Flip u,v for positive Jacobian
double u = interpolated_u_nst(s,0);
double v = interpolated_u_nst(s,1);
double w = interpolated_u_nst(s,2);

 //double u_exact = soln[0];
//double v_exact = soln[1];
//double w_exact = soln[2];

//double u_error = u-u_exact;
//double v_error = v-v_exact;
//double w_error = w-w_exact;

//arclength
double kk = Global_Physical_Variables::kappa;
double ta = Global_Physical_Variables::tau;

double Rc = Global_Physical_Variables::Rc;
double P = Global_Physical_Variables::Pitch;
double Ppi = P/(8.0*atan(1.0));

double AA = Global_Physical_Variables::Arc_Angle;

//double g1 = Global_Physical_Variables::g1;
double g12 = Global_Physical_Variables::g12;
double g2 = Global_Physical_Variables::g2;

double al = g*(AA)/nlayer;

//Initialise local vel/co

double X,Y,L;
double U,V,W;

//Straight Pipe bit

if (al<g12){

//cout << "al is" << al << endl;
//cout << "doing 1" << endl;
```

```
X = Rc - x;
Y = -y*ta + kk*z;
L = sqrt(Rc*Rc + Ppi*Ppi)*al;

W = kk*v + ta*w;
U = -u;
V = -ta*v + kk*w;
}

//Helical Bit
if (al>=g12 && al<g2){

//cout << "al is" << al << endl;
//cout << "doing 2" << endl;

X = Rc*(1.0+g12*sin(al-g12)) - x*cos(al-g12) - y*sin(al-g12);
Y = x*ta*sin(al-g12) - y*ta*cos(al-g12) +
kk*z - ta*Rc*al+ta*cos(al-g12)*Rc*g12;
L = sqrt(Rc*Rc + Ppi*Ppi)*al;

W = u*-kk*sin(al-g12) + v*kk*cos(al-g12) + w*ta;
U = -u*cos(al-g12) - v*sin(al-g12);
V = ta*sin(al-g12)*u - ta*cos(al-g12)*v+kk*w;
}

//End Straight Bit
if (al>=g2){
//cout << "al is" << al << endl;
//cout << "doing 3" << endl;

X = Rc+Rc*g12*sin(g2-g12) - x*cos(g2-g12) - y*sin(g2-g12);
Y = x*ta*sin(g2-g12) - y*ta*cos(g2-g12) +
kk*z - ta*Rc*g2 + ta*Rc*g12*cos(g2-g12);
L = sqrt(Rc*Rc + Ppi*Ppi)*al;

W = -kk*sin(g2-g12)*u + kk*cos(g2-g12)*v + ta*w;
U = -cos(g2-g12)*u - sin(g2-g12)*v;
V = ta*sin(g2-g12)*u - ta*cos(g2-g12)*v + kk*w;
}

// Output the global coordinates:
```

```
outfile << x << " " << y << " " << z << " " ;
outfile << u << " " << v << " " << w << " " ;
//outfile << u_exact << " " << v_exact
 << " " << w_exact << " " ;
//outfile << u_error << " " << v_error
 << " " << w_error << " " ;
// Output the local coordinates:
outfile << X << " " << Y << " "
<< al << " " << L << " ";
outfile << U << " " << V << " " << W
 << " " << V*V+U*U << " " ;
// Output the mesh layer:
outfile << g << " " ;
// Pressure
outfile << this->interpolated_p_nst(s)  << " " ;

outfile << std::endl;
}
outfile << std::endl;


// Write tecplot footer (e.g. FE connectivity lists)
this->write_tecplot_zone_footer(outfile,nplot);
}

    };
}

//=start_of_MyHelicalCylinder=====
///The arguemts are the radius of the tube,
///its curvature in the x,y plane
///and the pitch of the helix
//=====================
class MyHelicalCylinder : public GeomObject
{
public:

/// Constructor
 MyHelicalCylinder(const double& radiusa, const double& radiusb,
const double& rc, const double& pitch, const double& w,
const double& kap, const double& tau, const double& taubar) :
```

```
  GeomObject(3,3),RadiusA(radiusa),RadiusB(radiusb),Rc(rc),P(pitch),
W(w),Kappa(kap),Tau(tau),TauBar(taubar)
{
//Set the value of pi
Pi = 4.0*atan(1.0);
}

/// Destructor
virtual~MyHelicalCylinder(){}

///Lagrangian coordinate xi
///x0 - centreline, x1 - theta
//x2 - [0,1] makes volume instead of surface
void position (const Vector<double>& xi, Vector<double>& r) const
{
//double kap = Rc/sqrt(Rc*Rc+P*P/(4.0*Pi*Pi));
//double tau = (P/(2.0*Pi))/sqrt(Rc*Rc+P*P/(4.0*Pi*Pi));
//double taubar = (P/(2.0*Pi))/(Rc*Rc+P*P/(4.0*Pi*Pi));
//double taubar = 0.0;

double g = xi[0];
double g1 = Global_Physical_Variables::g1;
double g11 = Global_Physical_Variables::g11;
double g12 = Global_Physical_Variables::g12;
double g2 = Global_Physical_Variables::g2;
double g21 = Global_Physical_Variables::g21;
double g22 = Global_Physical_Variables::g22;
double x=0.0;
double y=0.0;
double z=0.0;

double smooth,sig1t,sig2t;

double sig1c = 0.5*xi[2]*cos(xi[0]*W)*cos(xi[1]) +
0.5*xi[2]*sin(xi[0]*W)*sin(xi[1]);

double sig2c = -0.5*xi[2]*sin(xi[0]*W)*cos(xi[1]) +
0.5*xi[2]*cos(xi[0]*W)*sin(xi[1]);

double sig1s = RadiusA*xi[2]*cos(xi[0]*W)*cos(xi[1]) +
RadiusB*xi[2]*sin(xi[0]*W)*sin(xi[1]);
```

```
double sig2s = -RadiusA*xi[2]*sin(xi[0]*W)*cos(xi[1]) +
RadiusB*xi[2]*cos(xi[0]*W)*sin(xi[1]);

double sig1h = RadiusA*xi[2]*cos(xi[0]*W)*cos(xi[1]) +
RadiusB*xi[2]*sin(xi[0]*W)*sin(xi[1]);

double sig2h = -RadiusA*xi[2]*sin(xi[0]*W)*cos(xi[1]) +
RadiusB*xi[2]*cos(xi[0]*W)*sin(xi[1]);

double sig1e = RadiusA*xi[2]*cos(xi[0]*W)*cos(xi[1]) +
RadiusB*xi[2]*sin(xi[0]*W)*sin(xi[1]);

double sig2e = -RadiusA*xi[2]*sin(xi[0]*W)*cos(xi[1]) +
RadiusB*xi[2]*cos(xi[0]*W)*sin(xi[1]);

if (g < g1) {

x = Rc+sig1c;
y = Rc*g - Tau*sig2c;
z = P/(2.0*Pi)*g + Kappa*sig2c;
}


if (g>=g1 && g < g11) {

smooth = 6*pow((g-g1)/(g11-g1),5) -
15*pow((g-g1)/(g11-g1),4) + 10*pow((g-g1)/(g11-g1),3);

sig1t = (smooth*RadiusA + 0.5*(1-smooth))*xi[2]*
cos(xi[0]*W)*cos(xi[1]) +
(smooth*RadiusB + 0.5*(1-smooth))*xi[2]*sin(xi[0]*W)*sin(xi[1]);

sig2t = -(smooth*RadiusA + 0.5*(1-smooth))*xi[2]*
sin(xi[0]*W)*cos(xi[1]) +
(smooth*RadiusB + 0.5*(1-smooth))*xi[2]*cos(xi[0]*W)*sin(xi[1]);

x = Rc+sig1t;
y = Rc*g - Tau*sig2t;
z = P/(2.0*Pi)*g + Kappa*sig2t;
}
```

```
if (g>=g11 && g<g12){

x = Rc+sig1s;
y = Rc*g - Tau*sig2s;
z = P/(2.0*Pi)*g + Kappa*sig2s;

}

if (g>=g12 && g<g2) {
x = Rc*cos(g-g12) + sig1h*cos(g-g12) + sig2h*Tau*sin(g-g12);
y = Rc*g12 + Rc*sin(g-g12) + sig1h*sin(g-g12) -
sig2h*Tau*cos(g-g12);
z = P/(2.0*Pi)*g + Kappa*sig2h;
}

if (g>=g2 && g<g21) {
x = Rc*cos(g2-g12) - Rc*sin(g2-g12)*(g-g2) +
cos(g2-g12)*sig1e + sin(g2-g12)*Tau*sig2e;
y = Rc*g12 + Rc*sin(g2-g12) + Rc*cos(g2-g12)*(g-g2) -
cos(g2-g12)*Tau*sig2e + sin(g2-g12)*sig1e;
z = P/(2.0*Pi)*g + Kappa*sig2e;
}

if (g>=g21 && g < g22) {

smooth = 6*pow((g-g21)/(g22-g21),5) -
15*pow((g-g21)/(g22-g21),4) + 10*pow((g-g21)/(g22-g21),3);

sig1t = (smooth*0.5 + RadiusA*(1-smooth))*xi[2]*
cos(xi[0]*W)*cos(xi[1]) +
(smooth*0.5 + RadiusB*(1-smooth))*xi[2]*sin(xi[0]*W)*sin(xi[1]);

sig2t = -(smooth*0.5 + RadiusA*(1-smooth))*xi[2]*
sin(xi[0]*W)*cos(xi[1]) +
(smooth*0.5 + RadiusB*(1-smooth))*xi[2]*cos(xi[0]*W)*sin(xi[1]);

x = Rc*cos(g2-g12) - Rc*sin(g2-g12)*(g-g2) +
cos(g2-g12)*sig1t + sin(g2-g12)*Tau*sig2t;
y = Rc*g12 + Rc*sin(g2-g12) + Rc*cos(g2-g12)*(g-g2) -
cos(g2-g12)*Tau*sig2t + sin(g2-g12)*sig1t;
```

```
z = P/(2.0*Pi)*g + Kappa*sig2t;
}

if (g >= g22) {

x = Rc*cos(g2-g12) - Rc*sin(g2-g12)*(g-g2) +
cos(g2-g12)*sig1c + sin(g2-g12)*Tau*sig2c;
y = Rc*g12 + Rc*sin(g2-g12) + Rc*cos(g2-g12)*(g-g2) -
cos(g2-g12)*Tau*sig2c + sin(g2-g12)*sig1c;
z = P/(2.0*Pi)*g + Kappa*sig2c;
}

r[0] = x;
r[1] = y;
r[2] = z;

//r[0] = (1.0/Delta)*cos(xi[0]) - cos(xi[0])*(RadiusA*xi[2]*
cos(xi[0]*W)*cos(xi[1]) + RadiusB*xi[2]*sin(xi[0]*W)*sin(xi[1])) -
P*sin(xi[0])*(RadiusA*xi[2]*sin(xi[0]*W)*cos(xi[1]) -
RadiusB*xi[2]*cos(xi[0]*W)*sin(xi[1]))/sqrt((1.0/Delta)*
(1.0/Delta)*Pi*Pi*4+P*P);

//r[1] = (1.0/Delta)*sin(xi[0]) - sin(xi[0])*(RadiusA*xi[2]*
cos(xi[0]*W)*cos(xi[1]) + RadiusB*xi[2]*sin(xi[0]*W)*sin(xi[1])) +
P*cos(xi[0])*(RadiusA*xi[2]*sin(xi[0]*W)*cos(xi[1]) - RadiusB*xi[2]*
cos(xi[0]*W)*sin(xi[1]))/sqrt((1.0/Delta)*(1.0/Delta)*Pi*Pi*4+P*P);

//r[2] = P*xi[0]/(2.0*Pi) - 2*Pi*(1.0/Delta)*(RadiusA*xi[2]*
sin(xi[0]*W)*cos(xi[1]) - RadiusB*xi[2]*cos(xi[0]*W)*sin(xi[1]))/
sqrt((1.0/Delta)*(1.0/Delta)*Pi*Pi*4+P*P);

//r[0] = (1.0/Delta)*cos(xi[0]) + xi[2]*Radius*cos(xi[0])*
cos(xi[1]) + P*xi[2]*Radius*sin(xi[0])*sin(xi[1])/(sig2);
//r[1] = (1.0/Delta)*sin(xi[0]) + xi[2]*Radius*sin(xi[0])*
cos(xi[1]) - P*xi[2]*Radius*cos(xi[0])*sin(xi[1])/(sig2);
//r[2] = P*xi[0]/(2.0*Pi) + 2*Pi*(1.0/Delta)*xi[2]*
Radius*sin(xi[1])/(sig2);
}


/// Return the position of the tube as a function of time
```

```
/// (doesn't move as a function of time)
void position(const unsigned& t,
const Vector<double>& xi, Vector<double>& r) const
{
position(xi,r);
}

private:
double Pi;
//double Radius;
double RadiusA;
double RadiusB;
double Rc;
double P;
double W;
double Tau;
double Kappa;
double TauBar;
};



//=start_of_problem_class=============
/// Entry flow problem in tapered tube domain
//============================
template<class ELEMENT, class TIMESTEPPER>
class UnsteadyHelicalProblem : public Problem
{

public:

/// Constructor: Pass DocInfo object and target errors
UnsteadyHelicalProblem(const double& min_error_target,
const double& max_error_target);

//DocInfo& doc_info); //, const double& min_error_target,
//const double& max_error_target);

/// Destructor (empty)
~UnsteadyHelicalProblem() {}
```

```
/// \short Update the problem specs before solve
void actions_before_newton_solve() {};

/// After adaptation: Pin redudant pressure dofs.
void actions_after_adapt()
{
// Pin redudant pressure dofs
RefineableNavierStokesEquations<3>::
pin_redundant_nodal_pressures(mesh_pt()->
element_pt());
}

//Actions After Distriubte - same as Actions after_adapt

void actions_after_distribute()
{
// Pin redudant pressure dofs
RefineableNavierStokesEquations<3>::
pin_redundant_nodal_pressures(mesh_pt()->element_pt());
}


//Actions before timestep: Update no slip on upper
//oscillating wall
void actions_before_implicit_timestep();

/// Run an unsteady simulation
void unsteady_run(DocInfo& doc_info);

/// Doc the solution
void doc_solution(DocInfo& doc_info, double dt);

/// \short Dump problem data to allow for later restart
void dump_it(DocInfo& doc_info, double dt);

/// \short Read problem data for restart
void restart(ifstream& restart_file);

void restartmpi();

/// Create a function to doc the mesh adaptation
```

```
//void doc_mesh();

/// \short Set initial condition (incl previous timesteps)
/// according to specified function.
void set_initial_condition();


/// \short Overload generic access function by one that returns
/// a pointer to the specific  mesh
RefineableTubeMesh<ELEMENT>* mesh_pt()
{
return dynamic_cast<RefineableTubeMesh<ELEMENT>*>
(Problem::mesh_pt());
}

private:

/// Exponent for bluntness of velocity profile
//int Alpha;

/// Doc info object Passed in by constructor
//DocInfo Doc_info;

///Pointer to GeomObject that specifies the domain boundary
GeomObject*Wall_pt;

}; // end_of_problem_class




//=start_of_constructor======================
/// Constructor: Pass DocInfo object and error targets
//====================================
template<class ELEMENT,class TIMESTEPPER>
UnsteadyHelicalProblem<ELEMENT,TIMESTEPPER>::UnsteadyHelicalProblem
(const double& min_error_target,const double& max_error_target)

/*DocInfo& doc_info),
const double& min_error_target,
const double& max_error_target)*/
```

```
 //: Doc_info(doc_info)

{  //Allocate the timestepper
add_time_stepper_pt(new TIMESTEPPER);

//Increase the value of the maximum residuals so that the first
//newton step converges.
Max_residuals = 100.0;

// Setup mesh:
//------------

//Build geometric object that forms the domain boundary:
//a tapered curved tube
double rcc=Global_Physical_Variables::Rc;
double p=Global_Physical_Variables::Pitch;
double ww=Global_Physical_Variables::W;
double ra = Global_Physical_Variables::RadiusA;
double rb = Global_Physical_Variables::RadiusB;
double kk = Global_Physical_Variables::kappa;
double tt = Global_Physical_Variables::tau;
double ttb = Global_Physical_Variables::taubar;

// Create GeomObject that specifies the domain boundary
Wall_pt=new MyHelicalCylinder(ra,rb,rcc,p,ww,kk,tt,ttb);
//Define pi
const double pi = 4.0*atan(1.0);

//Set the centerline coordinates for the start and end of the helix
Vector<double> centreline_limits(2);
centreline_limits[0] = 0.0;
centreline_limits[1] = Global_Physical_Variables::Arc_Angle;

//Set the positions of the angles that divide
//the outer ring of elements
Vector<double> theta_positions(4);
theta_positions[0] = -0.75*pi;
theta_positions[1] = -0.25*pi;
theta_positions[2] = 0.25*pi;
theta_positions[3] = 0.75*pi;
```

```
//Define the radial fraction for the central box
//in the domain
Vector<double> radial_frac(4,0.5);

// Number of layers in the mesh
unsigned nlayer=Mesh_Parameters::nlayer;

// Build and assign mesh
Problem::mesh_pt()=
new RefineableTubeMesh<ELEMENT>(Wall_pt,
centreline_limits,
theta_positions,
radial_frac,
nlayer, time_stepper_pt());

// Set error estimator - not sure if needed
Z2ErrorEstimator* error_estimator_pt=
new Z2ErrorEstimator;

mesh_pt()->spatial_error_estimator_pt()=
error_estimator_pt;

// Error targets for adaptive refinement
mesh_pt()->max_permitted_error()=
Global_Physical_Variables::MaxTol;

mesh_pt()->min_permitted_error()=
Global_Physical_Variables::MinTol;

// Set the boundary conditions for this problem:
//All nodal values are free by default
// -- just pin the ones that have Dirichlet conditions here.
//Choose the conventional form by setting gamma to zero
//The boundary conditions will be pseudo-traction free (d/dn = 0)
ELEMENT::Gamma[0] = 0.0;
ELEMENT::Gamma[1] = 0.0;
ELEMENT::Gamma[2] = 0.0;

// Output the geometry:
// doc_mesh();
```

```
unsigned num_bound = mesh_pt()->nboundary();

for(unsigned ibound=0;ibound<num_bound;ibound++)
{

unsigned num_nod= mesh_pt()->nboundary_node(ibound);

for (unsigned inod=0;inod<num_nod;inod++)
{
// Boundary 0 is the inlet symmetry boundary:
// Boundary 1 is the tube wall
// Pin all values
if((ibound==0) || (ibound==1))
{
mesh_pt()->boundary_node_pt(ibound,inod)->pin(0);
mesh_pt()->boundary_node_pt(ibound,inod)->pin(1);
mesh_pt()->boundary_node_pt(ibound,inod)->pin(2);
}

//Boundary 2 is the outflow boundary, pin z,
//leave traction  free
if(ibound==2)
{
//mesh_pt()->boundary_node_pt(ibound,inod)->pin(2);
}
}
} // end loop over boundaries


// Global_Physical_Variables::ReSt =
Global_Physical_Variables::Re *
Global_Physical_Variables::St;

// Loop over the elements to set up element-specific
// things that cannot be handled by constructor
unsigned n_element = mesh_pt()->nelement();
for(unsigned i=0;i<n_element;i++)
{
// Upcast from GeneralisedElement to the present element
```

```
ELEMENT* el_pt = dynamic_cast<ELEMENT*>
(mesh_pt()->element_pt(i));


//Set the Reynolds number, etc
el_pt->re_pt() = &Global_Physical_Variables::Re;

//cout << "The value of ReSt is " <<
Global_Physical_Variables::ReSt <<
" thanks " << std::endl;

//Set the Wormesly number
el_pt->re_st_pt() = &Global_Physical_Variables::ReSt;

}

// Pin redudant pressure dofs
RefineableNavierStokesEquations<3>::
pin_redundant_nodal_pressures(mesh_pt()->element_pt());

//Attach the boundary conditions to the mesh
cout <<"Number of equations: " << assign_eqn_numbers() << std::endl;

} // end_of_constructor

//============start_of_set_initial_condition=====
/// \short Set initial condition: Assign previous and
/// current values from exact solution.
/// Borrowed from Rayleigh,
//==================================
template<class ELEMENT,class TIMESTEPPER>
void UnsteadyHelicalProblem<ELEMENT,TIMESTEPPER>::
set_initial_condition()

{
//cout << "***************" << std::endl;
//cout << " SETTING INITIAL CONDITION" << std::endl;
//cout << "***************" << std::endl;

// Read restart data:
//-------------------
```

```
if (Global_Physical_Variables::checkRestart==1)
{
// Read the data from restart file and find out if the restart file
// was from an unsteady run
//restart(*restart_file_pt);

//cout << "************************" << std::endl;
//cout << " STARTING FROM A RESTART " << std::endl;
//cout << "************************" << std::endl;

restartmpi();

}
// Assign initial condition from exact solution
//-----------------------------------------

else
{

//cout << "********************" << std::endl;
//cout << " ASSIGNING A 0 START " << std::endl;
//cout << "********************" << std::endl;


//Find number of nodes in mesh
unsigned num_nod = mesh_pt()->nnode();

// Loop over the nodes to set initial guess everywhere
for (unsigned n=0;n<num_nod;n++)
{

// Assign solution

mesh_pt()->node_pt(n)->set_value(0,0.0);//soln[0]);
mesh_pt()->node_pt(n)->set_value(1,0.0);//soln[2]);
mesh_pt()->node_pt(n)->set_value(2,0.0);
}

}

//actions_before_implicit_timestep();
```

```
} // end of set_initial_condition


//=start_of_actions_before_implicit_timestep===
/// Set the inflow boundary conditions
//==============================
template<class ELEMENT,class TIMESTEPPER>

void UnsteadyHelicalProblem<ELEMENT,TIMESTEPPER>::
actions_before_implicit_timestep()

{

// (Re-)assign velocity profile at inflow values
//-----------------------------------------

//Get time
double time = time_pt()->time();

//Boundary Condition 1 - Wormersely

if (Global_Physical_Variables::boundary_parameter==1){


// (Re-)assign velocity profile at inflow values
//-----------------------------------------
Vector<double> xe(3);
Vector<double> soln(3);

unsigned ibound=0;
unsigned num_nod= mesh_pt()->nboundary_node(ibound);

for (unsigned inod=0;inod<num_nod;inod++)
{

// Recover coordinates
double x=mesh_pt()->boundary_node_pt(ibound,inod)->
x(0) - Global_Physical_Variables::Rc;

double y=mesh_pt()->boundary_node_pt(ibound,inod)->x(1);
```

```
double z=mesh_pt()->boundary_node_pt(ibound,inod)->x(2);

Vector<double> xe(3);
Vector<double> soln(3);

xe[0] = x;
xe[1] = y;
xe[2] = z;

//Set Velocity Components

ExactSoln::get_exact_u(time,xe,soln);

mesh_pt()->boundary_node_pt(ibound,inod)->set_value(1,soln[1]);

mesh_pt()->boundary_node_pt(ibound,inod)->set_value(0,0.0);

mesh_pt()->boundary_node_pt(ibound,inod)->set_value(2,soln[2]);
}
}

//Square wave - boundary 2

if (Global_Physical_Variables::boundary_parameter==2){

double f = 0.0;

double RadA = Global_Physical_Variables::RadiusA;
double RadB = Global_Physical_Variables::RadiusB;
double Kap = Global_Physical_Variables::kappa;
double Tau = Global_Physical_Variables::tau;

unsigned ibound=0;
unsigned num_nod= mesh_pt()->nboundary_node(ibound);

for (unsigned inod=0;inod<num_nod;inod++)
{

// Recover coordinates
```

```
double x=mesh_pt()->boundary_node_pt(ibound,inod)->
x(0) - Global_Physical_Variables::Rc;

double y=mesh_pt()->boundary_node_pt(ibound,inod)->x(1);

double z=mesh_pt()->boundary_node_pt(ibound,inod)->x(2);

double yandz2 = (y*y+z*z);
double r_R2=0.0;
double g1 = Global_Physical_Variables::g1;

if(g1<0){
r_R2 = (x*x)/(RadA*RadA) + (yandz2)/(RadB*RadB);
}

else if(g1>0){
r_R2 = (x*x)/(0.25) + (yandz2)/(0.25);
}

double PERIOD = Global_Physical_Variables::PERIOD;

//Set Velocity Components

ExactSoln::get_square(2*time/PERIOD,f);

double P_flow = 2.0*(1.0 - r_R2);

double yflow = Kap*P_flow*f;

double zflow = Tau*P_flow*f;

mesh_pt()->boundary_node_pt(ibound,inod)->set_value(1,yflow);

mesh_pt()->boundary_node_pt(ibound,inod)->set_value(0,0.0);

mesh_pt()->boundary_node_pt(ibound,inod)->set_value(2,zflow);

}
}

//Steady Flow - boundary 3
```

```
if (Global_Physical_Variables::boundary_parameter==3) {

double RadA = Global_Physical_Variables::RadiusA;
double RadB = Global_Physical_Variables::RadiusB;
double Kap = Global_Physical_Variables::kappa;
double Tau = Global_Physical_Variables::tau;

unsigned ibound=0;
unsigned num_nod= mesh_pt()->nboundary_node(ibound);

for (unsigned inod=0;inod<num_nod;inod++)
{
// Recover coordinates
double x=mesh_pt()->boundary_node_pt(ibound,inod)->
x(0) - Global_Physical_Variables::Rc;

double y=mesh_pt()->boundary_node_pt(ibound,inod)->x(1);

double z=mesh_pt()->boundary_node_pt(ibound,inod)->x(2);

double yandz2 = (y*y+z*z);
double r_R2=0.0;
double g1 = Global_Physical_Variables::g1;

if(g1<0){
r_R2 = (x*x)/(RadA*RadA) + (yandz2)/(RadB*RadB);
}

else if(g1>0){
r_R2 = (x*x)/(0.25) + (yandz2)/(0.25);
}

//Set Velocity Components

double P_flow = 2.0*(1.0 - r_R2);

double yflow = Kap*P_flow;

double zflow = Tau*P_flow;
```

```
mesh_pt()->boundary_node_pt(ibound,inod)->set_value(1,yflow);

mesh_pt()->boundary_node_pt(ibound,inod)->set_value(0,0.0);

mesh_pt()->boundary_node_pt(ibound,inod)->set_value(2,zflow);

}
}

//boundary 4 - suddenly stopping

if (Global_Physical_Variables::boundary_parameter==4) {

double RadA = Global_Physical_Variables::RadiusA;
double RadB = Global_Physical_Variables::RadiusB;
double Kap = Global_Physical_Variables::kappa;
double Tau = Global_Physical_Variables::tau;

unsigned ibound=0;
unsigned num_nod= mesh_pt()->nboundary_node(ibound);

for (unsigned inod=0;inod<num_nod;inod++)
{

// Recover coordinates

double x=mesh_pt()->boundary_node_pt(ibound,inod)->
x(0) - Global_Physical_Variables::Rc;

double y=mesh_pt()->boundary_node_pt(ibound,inod)->x(1);

double z=mesh_pt()->boundary_node_pt(ibound,inod)->x(2);

double yandz2 = (y*y+z*z);
double r_R2=0.0;
double g1 = Global_Physical_Variables::g1;

if(g1<0){
r_R2 = (x*x)/(RadA*RadA) + (yandz2)/(RadB*RadB);
}
```

```
else if(g1>0){
r_R2 = (x*x)/(0.25) + (yandz2)/(0.25);
}

double tstop = Global_Physical_Variables::t_stop;
double smooth = 0.0;
double tgap = 0.5;

//Set Velocity Components

if(time<tstop)
{
double P_flow = 2.0*(1.0 - r_R2);

double yflow = Kap*P_flow;

double zflow = Tau*P_flow;

mesh_pt()->boundary_node_pt(ibound,inod)->set_value(1,yflow);

mesh_pt()->boundary_node_pt(ibound,inod)->set_value(0,0.0);

mesh_pt()->boundary_node_pt(ibound,inod)->set_value(2,zflow);
}

else if(time>=tstop && time<tstop+tgap)
{

smooth = 6*pow((time-tstop)/(tgap),5) -
15*pow((time-tstop)/(tgap),4) + 10*pow((time-tstop)/(tgap),3);

double P_flow = (1-smooth)*2.0*(1.0 - r_R2);

double yflow = Kap*P_flow;

double zflow = Tau*P_flow;

mesh_pt()->boundary_node_pt(ibound,inod)->set_value(1,yflow);

mesh_pt()->boundary_node_pt(ibound,inod)->set_value(0,0.0);
```

```
mesh_pt()->boundary_node_pt(ibound,inod)->set_value(2,zflow);
}

else if(time>=tstop+tgap)
{
mesh_pt()->boundary_node_pt(ibound,inod)->set_value(1,0.0);

mesh_pt()->boundary_node_pt(ibound,inod)->set_value(0,0.0);

mesh_pt()->boundary_node_pt(ibound,inod)->set_value(2,0.0);
}
}
}

//Saw Tooth Wave

else if (Global_Physical_Variables::boundary_parameter==5){

// (Re-)assign velocity profile at inflow values
//-----------------------------------------

Vector<double> xe(3);
Vector<double> soln(3);

unsigned ibound=0;
unsigned num_nod= mesh_pt()->nboundary_node(ibound);

for (unsigned inod=0;inod<num_nod;inod++)
{

// Recover coordinates
double x=mesh_pt()->boundary_node_pt(ibound,inod)->
x(0) - Global_Physical_Variables::Rc;

double y=mesh_pt()->boundary_node_pt(ibound,inod)->x(1);

double z=mesh_pt()->boundary_node_pt(ibound,inod)->x(2);

Vector<double> xe(3);
Vector<double> soln(3);
```

```
xe[0] = x;
xe[1] = y;
xe[2] = z;

//Set Velocity Components

ExactSoln::get_sawtooth_u(time,xe,soln);

mesh_pt()->boundary_node_pt(ibound,inod)->set_value(1,soln[1]);

mesh_pt()->boundary_node_pt(ibound,inod)->set_value(0,0.0);

mesh_pt()->boundary_node_pt(ibound,inod)->set_value(2,soln[2]);
}
}


} // end_of_actions_before_implicit_timestep


//=start_of_doc_solution=============
/// Doc the solution
//============================
template<class ELEMENT, class TIMESTEPPER>

void UnsteadyHelicalProblem<ELEMENT,TIMESTEPPER>::
doc_solution(DocInfo& doc_info, double dt)
{

ofstream some_file;
char filename[100];

// Number of plot points
unsigned npts;
npts=5;

//Need high precision for large radii of curvature
//some_file.precision(10);

int my_rank = this->communicator_pt()->my_rank();
```

```
// Output solution
sprintf(filename,"%s/soln_t_%g_Re%g_ReSt%g_on_proc%i.dat",
doc_info.directory().c_str(), Global_Physical_Variables::timer,
Global_Physical_Variables::Re,Global_Physical_Variables::ReSt,
my_rank);

some_file.open(filename);

//cout << "time is" << doc_info.number()*dt << endl;

mesh_pt()->output(some_file,npts);
some_file.close();

/*
 // Doc error
 //----------
 double error,norm;
 sprintf(filename,"%s/error%i.dat",doc_info.directory().c_str(),
doc_info.number());

 some_file.open(filename);

 mesh_pt()->compute_error(some_file,
ExactSoln::get_exact_u,
time_pt()->time(),
error,norm);

some_file.close();

*/

} // end_of_doc_solution

//=====start_of_restart====
/// Restart
//==================

template<class ELEMENT, class TIMESTEPPER>

void UnsteadyHelicalProblem<ELEMENT,TIMESTEPPER>::restartmpi()
```

```
{

// Get current processor rank:
int my_rank=this->communicator_pt()->my_rank();
int nproc=this->communicator_pt()->nproc();

char filename[100];
sprintf(filename,"%s/Restart_t_%g_%iproc%i_DUMP.dat",
Global_Physical_Variables::results_folder,
Global_Physical_Variables::Restart_Time,
nproc,my_rank);

// Pointer to restart file
ifstream* restart_file_pt=0;

// Open restart file from stem
restart_file_pt= new ifstream(filename,ios_base::in);

if (restart_file_pt!=0)
{
oomph_info << "Have opened " << filename <<
" for restart. " << std::endl;
}

else
{
std::ostringstream error_stream;
error_stream
<< "ERROR while trying to open " << filename
<< " for restart." << std::endl;

throw OomphLibError(
error_stream.str(),
"UnsteadyHelicalProblem<ELEMENT,TIMESTEPPER>::restartmpi()",
OOMPH_EXCEPTION_LOCATION);
}

// Read restart data:
//-------------------

if (restart_file_pt!=0)
```

```
{

// Read the data from restart file and find out if the restart file
// was from an unsteady run

oomph_info << "Restarting..." << std::endl;
UnsteadyHelicalProblem::restart(*restart_file_pt);

}

} // end of restart

//=start_of_doc_mesh=====
/// Doc the mesh
//==================

/*template<class ELEMENT>

void SteadyHelicalProblem<ELEMENT>::doc_mesh()

{
ofstream yet_another_file;
char filename[200];

// Number of plot points
unsigned npts;
npts=3; ///

// Set up doc info
DocInfo doc_info;
// Set output directory
doc_info.set_directory("MeshGen");

// Get current processor rank:
int my_rank=this->communicator_pt()->my_rank();
int nproc=this->communicator_pt()->nproc();

cout << "\n\n" << std::endl;
oomph_info <<"==> Documenting solution after adaptation
number 1 " << "\n" << std::endl;
```

```
sprintf(filename,"%s/%s_L%g_Re%g_%iproc%i_MESH_adapt_no%i.dat",
doc_info.directory().c_str(),"MeshCords",
Global_Physical_Variables::Length,Global_Physical_Variables::Re,
nproc,my_rank);

yet_another_file.open(filename);

mesh_pt()->output(yet_another_file,npts);

yet_another_file.close();

} // end_of_doc_mesh */


//=======start_of_dump_it====
/// Dump the solution to disk
//======================

template<class ELEMENT, class TIMESTEPPER>

void UnsteadyHelicalProblem<ELEMENT,TIMESTEPPER>::
dump_it(DocInfo& doc_info, double dt)

{
// Dump the refinement pattern and the generic problem data

ofstream dump_file;
char filename[200];

// Get current processor rank:
int my_rank=this->communicator_pt()->my_rank();
int nproc=this->communicator_pt()->nproc();

sprintf(filename,"%s/Restart_t_%g_%iproc%i_DUMP.dat",
doc_info.directory().c_str(),Global_Physical_Variables::timer,
nproc,my_rank);

dump_file.open(filename);

Problem::dump(dump_file);
dump_file.close();
```

```
} // end of dump_it

//=========start_of_restart=======
/// Read solution from disk
//=========================

template<class ELEMENT,class TIMESTEPPER>

void UnsteadyHelicalProblem<ELEMENT,TIMESTEPPER>::
restart(ifstream& restart_file)

{

// Refine the mesh and read in the generic problem data
Problem::read(restart_file);

} // end of restart

//===start_of_unsteady_run===========
/// Unsteady run...
//============================

template<class ELEMENT,class TIMESTEPPER>

void UnsteadyHelicalProblem<ELEMENT,TIMESTEPPER>::
unsteady_run(DocInfo& doc_info)

{

// Open trace file

/*char filename[100];

sprintf(filename,"%s/trace.dat",doc_info.directory().c_str());
Trace_file.open(filename);

// Write tecplot header for trace file
Trace_file << "time" << ",      "
<< "x" << ",         "
<< "y" << ",         "
```

```
<< "u_1" << ",       "
<< "u_2" << ",       "
<< "u_exact_1" << ", "
<< "u_exact_2" << ", "
<< "error_1" << ",   "
<< "error_2" << ",   "
<< "L2 error" << ",  "
<< "L2 norm" << ",   " << std::endl;
*/

 //Set value of dt
 double dt = Global_Physical_Variables::dt;

 //always start impulsively

// Initialise all history values for an impulsive start
assign_initial_values_impulsive(dt);
//cout << "IC = impulsive start" << std::endl;

//Now do many timesteps
unsigned ntsteps=Global_Physical_Variables::nsteps;

//first time step
bool first=true;

//set maximum adaptations, allow for lots and lots and
//lots and lots in the first time step
 unsigned max_adapt=10;

set_initial_condition();

// If restart: The first step isn't really the first step,
// i.e. initial condition should not be re-set when
// adaptive refinement has been performed. Also, limit
// the max. number of refinements per timestep to the
// normal value straightaway.

if (Global_Physical_Variables::checkRestart==1)
{
first=false;
max_adapt=2;
```

```
cout << "Starting From A Restart File" << std::endl;
}

// If no restart, refine mesh uniformly before we get started

else
{
refine_uniformly();

//refine_uniformly();

// Solution is automatically interpolated from the
//coarse initial mesh onto the refined mesh but
//this provides a very poor representation
// of the intial condition:
//Re-assign the initial conditions

set_initial_condition();
}

//refine_uniformly();

//Reset Time

//time_pt()->time() = 0.0;

actions_before_implicit_timestep();
// Doc initial condition
doc_solution(doc_info,dt);

// increment counter
doc_info.number()++;


double real_time = 0.0;

cout << "**********************" << std::endl;
cout << "THE VALUE OF MAX TOL IS " << std::endl;
cout << Global_Physical_Variables::MaxTol << std::endl;

//actions_before_implicit_timestep();
```

```
//Loop over the timesteps

for(unsigned t=1;t<=ntsteps;t++)
{

cout << "TIMESTEP " << t << std::endl;

//Take one fixed timestep
unsteady_newton_solve(dt,max_adapt,first);

//No longer the first run
first=false;

max_adapt=2;

real_time = time_pt()->time();

//Output the time
cout << "Time is now " << real_time << std::endl;
//cout << "First run check "

Global_Physical_Variables::timer = real_time;

cout << "timer value is now " <<
Global_Physical_Variables::timer << std::endl;

// Doc solution
doc_solution(doc_info,dt);

// increment counter
doc_info.number()++;

cout << "*****************" << std::endl;
cout << "*****************" << std::endl;
cout << "THE VALUE OF t mod 10 is " <<
 t%10 << " *************" << std::endl;
cout << "*****************" << std::endl;

if ( t % 10 == 0 )
{
```

```
dump_it(doc_info,dt);

cout << "************" << std::endl;
cout << "Writing Restart" << std::endl;
}

}

} // end of unsteady run

///////////////
///////////////
///////////////

//=start_of_main==========
/// Driver for 3D entry flow into a tapered tube.
///If there are any command line arguments,
///we regard this as a validation run
/// and perform only a single adaptation
//=====================

int main(int argc, char* argv[])
{

#ifdef OOMPH_HAS_MPI
// Initialise MPI
MPI_Helpers::init(argc,argv);
#endif

time_t current = time(0);
cout << ctime(&current)<< endl;

// Store command line arguments
CommandLineArgs::setup(argc,argv);

DocInfo mesh_doc_info;
bool report_stats=true;
mesh_doc_info.set_directory("RESLT_MESH");
mesh_doc_info.number()=10;

// Allow (up to) two rounds of fully automatic
```

```
// adapation in response to
//---------------------
// error estimate

//---------------

//Get the variables

Global_Physical_Variables::get_variables();

Mesh_Parameters::get_nlayer();

//unsigned max_adapt;
double max_error_target,min_error_target;

// Set max number of adaptations in black-box Newton solver and
// error targets for adaptation

//This is all turned off for unsteady work

max_error_target=Global_Physical_Variables::MaxTol;
min_error_target=Global_Physical_Variables::MinTol;

cout << "Error Tolerances Max " <<
max_error_target << " Min " << min_error_target << std::endl;

//std::cout << Global_Physical_Variables::RadiusA << " "
<< Global_Physical_Variables::Rc;

// Set up doc info

 DocInfo doc_info;

if (Global_Physical_Variables::checkRestart==0){
doc_info.number()=0;
}

// Do Taylor-Hood elements
//------------------------

// Set output directory
```

```
doc_info.set_directory(Global_Physical_Variables::results_folder);

// Step number

if (Global_Physical_Variables::checkRestart==0){
 doc_info.number()=0;
}

// Build problem

UnsteadyHelicalProblem<CustomisedRefineableQTaylorHoodElement,BDF<2> >
problem(min_error_target,max_error_target);


#ifdef OOMPH_HAS_MPI

problem.distribute();

#endif

if (Global_Physical_Variables::checkRestart==1)
{
oomph_info << "Loading restart file" << std::endl;

//problem.restartmpi(doc_info);
oomph_info << "Finished restart\n\n" << std::endl;

cout << "Starting from a Restart File" << std::endl;
}

//Reset Time

cout << " Doing Taylor-Hood elements " << std::endl;

// Solve the problem
problem.unsteady_run(doc_info);

current = time(0);
cout << ctime(&current)<< endl;

#ifdef OOMPH_HAS_MPI
```

```
MPI_Helpers::finalize();
#endif
} // end_of_main
```

Bibliography

Cunningham, F., Lenovo, K., Bloom, S., Hauth, J., Rouse, D. & C, S. (2009), *Williams Obstetrics*, 23rd edn, McGraw-Hill.

de Laat, M., Franx, A., van Alderen, E., Nikkels, P. & Visser, G. (2005), 'The umbilical coiling index, a review of literature', *Journal of Maternal-Fetal Neonatal Medicine* **17**(2), 93–100.

Dean, W. (1928), 'The stream-line motion of fluid in a curved pipe', *The London, Endiburgh, and Dublin Philosophical Magazine and Journal of Science* **5**(30), 673–695.

Dean, W. & Hurst, J. (1927), 'Note on the motion of fluid in a curved pipe', *The London, Endiburgh, and Dublin Philosophical Magazine and Journal of Science* **4**(20), 208–223.

Germano, M. (1982), 'On the effect of torsion on a helical pipe flow', *Journal of Fluid Mechanics* **125**, 1–8.

Germano, M. (1989), 'The dean equations extended to a helical pipe flow', *Journal of Fluid Mechanics* **203**, 289–305.

Ghezzi, F., Raio, L., Di Naro, E., Franchi, M., Balestreri, D. & V, D. (2001), 'Nomogram of Whatron's Jelly as depicted in the sonographic cross section of the umbilical cord', *Ultrasound in Obstetrics and Gynecology* **18**, 121–125.

Heil, M. & Hazel, A. (2017), 'oomph-lib'.
URL: *http://oomph-lib.maths.man.ac.uk/doc/html/index.html*

Jewell, N. & Denier, J. (2013), 'The decay of the flow in the end region of a suddenly blocked pipe', *Journal of Fluid Mechanics* **730**, 533–558.

Khong, T. (2010), 'Evidence-based pathology: umbilical cord coiling', *Pathology* **42**(7), 618–622.

Lekner, J. (2007), 'Viscous flow through pipes of various cross-sections', *European Journal of Physics* **28**, 521–527.

Levy, B., Denier, J. & Clark, A. (2015), An experimental investigation of the flow through a coiled tube.

Liu, S. & Masliyah, J. (1993), 'Axially invariant laminar flow in helical pipes with a finite pitch', *Journal of Fluid Mechanics* **251**, 315–353.

Manlapaz, R. & Churchill, S. (1980), 'Fully developed laminar flow in a helically coiled tube of finite pitch', *Chemical Enginnering Communications* **7**, 57–78.

Mazumdar, J. (1992), *Biofluid mechanics*, World Scientific Publishing Co. Pty. Ltd.

Morales, R. & Rosa, E. S. (2012), 'Modeling fully developed laminar flow in a helical duct with rectangular cross section and finite pitch', *Applied Mathematical Modelling* **36**, 5059–5067.

Nobari, M. & Malvandi, A. (2013), 'Torsion and curvature effects on fluid dlow in a helical annulus', *International Journal of Non-Linear Mechanics* **57**, 90–101.

Selvarasu, N. & Tafti, D. (2012), 'Investigation of the effects of dynamic change in curvature and torsion on pulsatile flow in a helical tube', *Journal of Biomechanical Engineering* **134**, 1–17.

Sherwin, S. & Blackburn, H. (2005), 'Three-dimensional instabilities and transition of steady and pulsatile axisymmetric stenotic flows', *Journal of Fluid Mechanics* **533**, 297–327.

Tuttle, E. (1990), 'Laminar flow in twisted pipes', *Journal of Fluid Mechanics* **219**, 545–570.

Vashisth, S., Kumar, V. & Nigam, K. (2008), 'A review on the potential applications of curved geometries in process industry', *Industrial and Engineering Chemistry Research* **47**(10), 3291–3337.

Wang, C. (1981), 'On the low-reynolds-number flow in a helical pipe', *Journal of Fluid Mechanics* **108**, 185–194.

Ward-Smith, A. (1980), *Internal fluid flow*, Oxford University Press.

Weerappuli, D. & Parakrama, V. (1987), 'Simulation of pulsatile flow in arteries using the finite-element method', *Retrospective Theses and Dissertations* (Paper 11657).

Wilke, D. (2016), Pressure and flow within the umbilical vessels, PhD thesis, The University of Adelaide.

Womersley, J. (1955), 'Method for the calculation of velocity, rate of flow and viscous drag in arteries when the pressure gradient is known', *Journal of Physiology* **127**(2), 553–563.

Zienkiewicz, O., Taylor, R. & Nithiarasu, P. (2014), *The Finite Element Method for Fluid Dynamics*, 7th edn, Butterworth-Heinemann.

Zienkiewicz, O., Taylor, R. & Zhu, J. (2013), *The Finite Element Method: Its Basis and Fundamentals*, seventh edn, Butterworth-Heinemann.