University of Nottingham
UK | CHINA | MALAYSIA

Model reduction for driven PDEs: Application to polymer constitutive equations

Thesis submitted to the University of Nottingham for the degree of **Doctor of Philosophy, March 2024.**

Daniel Mellor

10168389

Supervised by

Richard Stuart Graham
Matteo Icardi

Signature: 

Date: 31 / 3 / 2024

Abstract

In polymer dynamics, the direct derivation of equations for stress response is often very difficult due to complex dynamics arising from interactions between long-chain molecules. One useful approach is to map from an expensive molecular constitutive equation to a cheaper model using model reduction. There is an ever-growing need for computationally cheap polymer models as many important applications such as modelling polydispersity or computational fluid dynamics require a vast number of evaluations of the model. Currently, many of these applications use the Rolie-Poly model which has several known weaknesses. We develop a new data-driven way of carrying out model reduction for constitutive equations. This reduction is achieved by choosing a number of slow-moving variables as coarse-grained variables. A mapping between these coarse-grained variables and the full configuration is then created using a data-driven approach. With this mapping, we then evolve these coarse-grained variables, but not with directly derived differential equations. Instead, we utilise the mapping to map back to the full model and calculate derivatives using the original model. The result of this is a model that can take large timesteps but retains greater accuracy. Using this framework on the GLaMM model for polymer dynamics, we derive a new model reduction with the accuracy of the GLaMM model and the speed of the Rolie-Poly model, with the only minor limitation being the span of training data. This model is sufficiently fast to be adapted for use in many of the aforementioned applications.

Acknowledgements

First and foremost I would like to extend my sincerest thanks to my supervisors, Richard Graham and Matteo Icardi, for all the help and support they have provided me with over the years. The various topics we have discussed have undoubtedly led to not only the progression of my understanding of polymer dynamics but also my growth as a researcher. Their ideas and suggestions have been invaluable in shaping this thesis into its current form. It has been a pleasure to work with them.

I would like to express my deepest gratitude to my many colleagues who have worked alongside me during my PhD, particularly to Robert Barnett and Celine Lee who would always be willing to have lengthy discussions on how to overcome various problems. Similarly, I would like to thank Matthew Scase and Anna Kalogirou for taking the time to look over my work at various stages and offer constructive feedback.

I would like to thank the numerous people who made the Leeds workshop on Flow-Induced Crystallisation in Polymers possible. Producing polymer model predictions with the RepTate software furthered my understanding of various models within the field of polymer dynamics and the the presentations were valuable in gaining an insight into modern areas of research. Furthermore, I would like to thank the late Tom McLeish, one of the creators of the GLaMM model with whom I had the privilege of having a conversation at this workshop. The unique perspective he could provide due to his previous work led to the suggestion of several new innovative ideas for GLaMM model reduction during our discussion.

I would like to thank the many people who organised and gave presentations at the numerous online BSR conferences that I attended, which allowed me to remain up to date with many of the novel approaches being utilised in rheology, even throughout the pandemic. I would also like to express my appreciation to Tom Elgood and Kristoffer van der Zee for assis-

tance in navigating certain circumstances that arose due to the COVID-19 pandemic.

I would also like to thank Ron Larson for taking time out of his schedule during his visit to Nottingham to meet with me and discuss my work. During our conversation, he offered ideas to explore that I had not previously considered.

I also would like to acknowledge the funding from the EPSRC without which this work would not have been possible. Finally, I would like to thank each and every one of my friends and family at home for their help and support during my studies.

Contents

List of Figures

Chapter 1

Introduction

1.1 Model reduction for polymer dynamics

This thesis is concerned with model reduction, particularly for the field of polymer dynamics. A polymer is a substance comprised of very large macromolecules, each consisting of a chain of repeating subunits. This definition includes many plastics and resins which are ever-present in the modern world. Such materials are manufactured in liquid form with flow and temperature controlled to achieve the desired shape and rate of crystallisation. Understanding the dynamical response is therefore essential to produce polymer products. The difficulty arises since these liquids, commonly known as polymer melts, are non-Newtonian viscoelastic fluids that can act as viscous liquids or elastic solids depending on the flow imposed and the timescale of interest. This property is due to the size and shape of these macromolecules, with polymers structured as a long chain of repeating subunits. The stress response in a polymer melt is controlled by the shape that these molecules form under flow, and thus stress response is controlled by the chains' dynamics. These dynamics require specialised models

since the movement of an individual chain can be heavily constrained by its neighbours, as these chains entangle with each other. Any model for the dynamics of an entangled polymer melt must therefore account for these various constraints imposed by entanglements, and furthermore, how these constraints themselves are affected by the flow.

The accuracy of the model's predictions is an important factor when creating these models, but another consideration is the numerical cost. In a monodisperse entangled polymer melt, where all the chains are of the same length, under a uniform flow, evaluation with one of these polymer models at a single point is sufficient to extract the dynamics of the polymer melt. However, in polymer manufacture the polymers are often polydisperse, comprising of a distribution of various chain lengths and subject to more complex non-uniform flows. To model polydispersity we often need to solve multiple versions of the constitutive equations to account for different chain lengths and the interactions between them. This makes the model many times more expensive, even for simple flows. As for complex flows, the response is no longer the same for the entire flow field and requires computational fluid dynamics. Thus we discretise the flow field into many points to solve the Navier-Stokes equation numerically, requiring the constitutive equations to be solved at each of these locations. Both the effects of polydispersity and complex flows individually make modelling more numerically expensive, and when both are present the effect is compounded as every chain length and cross interaction must be evaluated at every point in the flow field. For this reason, both accuracy and low computational cost are required for utilisation under realistic conditions where one or both of these complications are often present.

1.2 Existing polymer models

One of the existing polymer models that has been shown to be close to experimental data in a range of situations is the GLaMM model [14]. In addition to its high accuracy, the GLaMM model has influenced a wide range of models in the field of polymer dynamics. The main drawback of the GLaMM model is its very high computational cost relative to other simpler constitutive equations. This restricts its practical use to monodisperse polymer melts in uniform flows which only covers a limited number of experiments, and excludes realistic polymer processing. For this reason, a simple model reduction of the GLaMM model known as the Rolie-Poly model [21] is often favoured instead. Owing to its simpler form, the Rolie-Poly model has a far lower computational cost, which has led to the Rolie-Poly model being adopted by many polymer manufacturers. It has also been used as the basis to create a polydisperse polymer model, the Rolie-Double-Poly model [6]. However, there are many cases in which the simple model reduction causes inaccurate predictions, which are also inherited by successive models such as the Rolie-Double-Poly model. A more accurate Rolie-Poly type model would be greatly beneficial but it is not clear how to directly extend the model reduction approach used in the Rolie-Poly paper.

1.3 Thesis overview

Our aim in this thesis is to create a reduced model for polymer dynamics with speed similar to the Rolie-Poly model, but greater accuracy, closer to the GLaMM model. To achieve this, we first produce a framework for data-driven model reduction that is applicable to a general PDE with an external driving term. For our aim we use the GLaMM model, but the

approach is applicable to other PDEs. This framework uses precomputed model data to map from selected slow-moving coarse-grained variables to the more detailed configuration of the PDE model. The idea of such an approach is that we evolve these few coarse-grained variables but do not do so by explicitly deriving differential equations for the coarse-grained variables. Instead, we map back to the more detailed full configuration, use the original PDE to calculate the derivative of this full configuration, and use this to calculate the derivative of the coarse-grained variables at every timestep. This significantly improves the accuracy of the evolution of the coarse-grained variables as the full PDE model is utilised to calculate the evolution. Since the time derivative of the full configuration is calculated at every step, the cost per timestep can see little improvement, but the major advancement comes from the fact that we are instead evolving the slow-moving coarse-grained variables. This allows us to take much larger timesteps, causing significant speed-up with the reduced number of evaluations whilst retaining accuracy. In this thesis, we manually select which quantities are used as our slow-moving coarse-grained variables, but it is possible to automate this process with machine learning, which we discuss in future work. By using this model reduction framework with the GLaMM model we created our reduced model for polymer dynamics.

This thesis covers the creation of the general framework, application to the GLaMM model and analysis of reduced model performance for uniaxial extension flow and shear flow prediction. Firstly, we review the relevant literature for previous polymer models and the current challenges in the field of polymer dynamics in chapter 2. Then in chapter 3 we establish the general framework, apply it to the GLaMM model, and analyse the performance of this new reduced model against Rolie-Poly predictions for uniaxial extension flows. In chapter 4 we use our model to predict shear

flows, the rotational element of which introduced a number of challenges. To ensure that our reduced model remains accurate for shear flow prediction, this chapter also introduces principal direction to the model along with natural generalisations to the definition of the coarse-grained variables and the mapping from these coarse-grained variables to the full configuration. Lastly, chapter 5 provides a final summary of our work, and re-examines the various graphical comparisons between our reduced model and the Rolie-Poly model to determine if we have sufficiently met our aim. This final chapter also covers a wide array of future avenues that are possible for this work such as automated selection of the coarse-grained variables for the general framework or applying our reduced model to more nuanced polymer melt qualities like polydispersity or flow-induced crystallisation.

All the code produced as part of this thesis is available on GitHub [28].

Chapter 2

Literature review

2.1 The Doi-Edwards tube model

A polymer is a material consisting of very large molecules each of which is comprised of a vast number of the same repeating unit, connected end-to-end to form long chains. The first hurdle to overcome in the modelling of polymer dynamics was how to feasibly calculate interactions within a polymer. Each chain will be entangled with many other chains, and thus to predict how a single chain will move, we require knowledge of the interactions from every surrounding chain, which would necessitate huge amounts of calculation. A widely accepted solution is the Doi-Edwards Tube model, created by Sam Edwards and Masao Doi in 1986 [8]. The tube model proposes that rather than evaluating each of the surrounding chains individually, we can use a mean-field approach to reduce the problem to that of the primitive path of a single chain within a finite "constraining" tube. The path is free to move within the tube, with the tube modelling the topological uncrossability with other chains. This is shown in figure 2.1. This reduces the many chain problem down to a single chain problem.

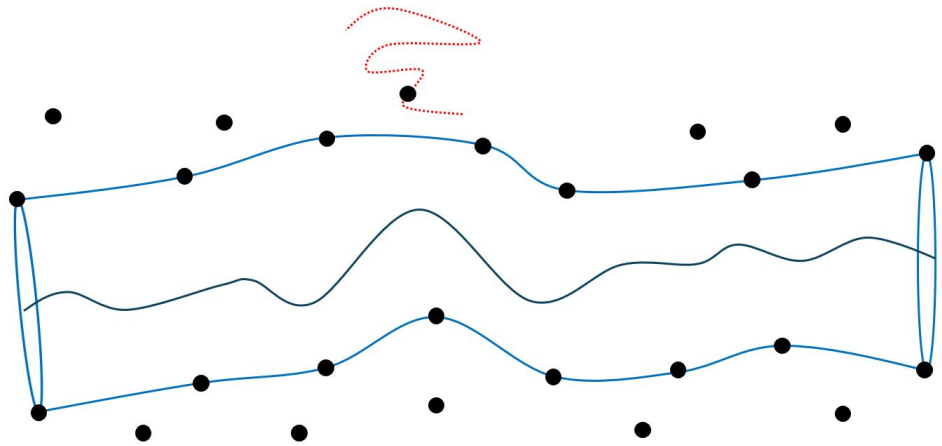


Figure 2.1: A 2D slice showing how the tube model replaces the topological effects of the individual entanglements with a constraining tube. Due to the sparsity of the chains in 3D space, each other chain will only contribute one or more points to this 2D slice of the tube, the points at which it intersects the current plane.

In manufacture polymers are often in liquid form where the polymer chains are free to move and interact with each other, often known as polymer melts. The polymers then crystallise over time when not exposed to high temperatures, becoming solid. This thesis will focus on the dynamics of polymer melts, as understanding their dynamics is key to achieving the desired final product. Due to the interaction between chains, these polymer melts exhibit viscoelastic behaviour. Initially, the polymer chains will have a random configuration known as the isotropic rest state where the chains within the polymer are completely without alignment and unstressed. If a flow is applied to a polymer in this state we will initially see an elastic response as the entanglements will resist the movement due to the flow. After some time, the chains will have had time to start relaxing and we transition to a behaviour akin to a viscous liquid, as the chains are now less restricted and free to "flow" past each other.

A flow will orient each polymer chain and its tube based on the direction of the flow. As time progresses, the polymer chains will try to relax back

towards the isotropic rest state. If we consider a polymer that has recently undergone a flow, causing strain, the tube model assumes that the polymer's stress is due to chains trapped in oriented tube segments. Thus if a portion of the polymer chain escapes the tube or a portion of the tube itself is removed that portion will lose some stress. There are several different methods by which an oriented polymer chain can relax. One such way is reptation, illustrated in figure 2.2.

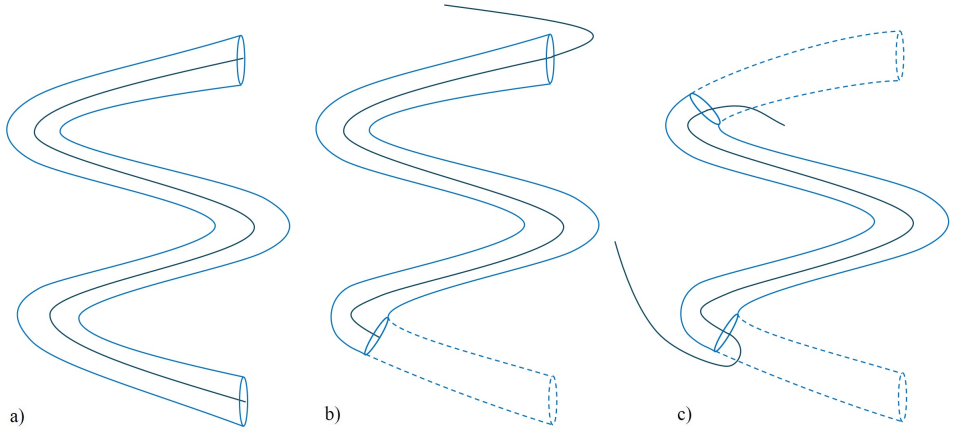


Figure 2.2: An illustration of relaxation by reptation after an imposed strain.

Reptation is diffusive motion of the entire chain, causing the polymer chain to move down the tube in a manner similar to a snake. We can see this in figure 2.2. Immediately after the imposed strain, the entirety of the chain is within the tube, depicted in (a). Diffusive motion of the polymer chain causes the segments of the tube that constrained the trailing end to be lost, as depicted in (b). The emerging end continually explores new randomly chosen tube segments, and since these are chosen with no constraint on their direction, they carry no stress. As the chain moves back and forth under diffusion, further segments are lost from both ends, as depicted in (c). Another way in which polymer chains can relax is contour length fluctuations (CLF), illustrated in figure 2.3.

The length of this primitive path within the tube is not fixed and this

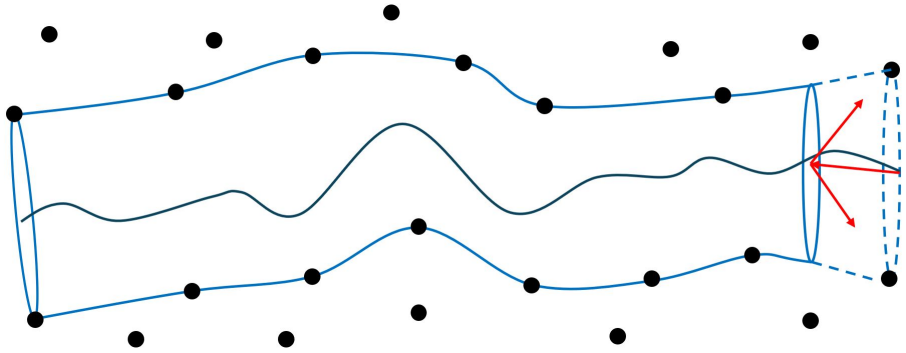


Figure 2.3: A figure to show how tube segments are lost to CLF.

contour length will fluctuate. This accelerates the rate at which the ends of the polymer chain explore new chain segments. If a section at the end of the tube temporarily contracts, the memory of the now empty tube segment is lost and when the section re-extends it is free to explore new tube segments. Relaxation still occurs during flow and the Weissenberg number is a measure of how quickly the flow is orienting the chains relative to the rate at which the chains are relaxing through the exploration of new tube segments. The effects of CLF are more significant on shorter chains.

A quantity that we need to introduce to use models that rely on the tube model is the dimensionless contour length, $Z \in \mathbb{R}^+$. To calculate this we divide total molecular weight by the weight between entanglement segments to get $Z = \frac{M_w}{M_e}$. Due to uncertainties in the measurements of these molecular weights, this value is then rounded so that $Z \in \mathbb{N}$, and Z can then be thought of as both the contour length and the number of entanglement segments. We also define the chain Rouse time, $\tau_R \in \mathbb{R}^+$, which is a material dependant quantity that controls the rate of relaxation of the chain contour length, and the reptation time, $\tau_d \approx 3Z\tau_R \in \mathbb{R}$, controlling the relaxation of the tube orientation due to reptation. Both the GLaMM model and Rolie-Poly model which we shall cover in detail in this literature review are based on the Doi-Edwards tube model.

2.2 Early ODE constitutive equations based on the tube model

The creation of the Doi-Edwards tube model was a significant step towards a model that could predict polymer dynamics, but there was still a significant disparity between many experimental results and model predictions. In particular, experimental results showed a far greater level of shear thinning than the Doi-Edwards model would otherwise suggest. By analysing the circumstances under which this divergence would occur, the theory of convective constraint release (CCR) was established, an additional mechanism by which a polymer chain can relax. This is illustrated in figure 2.4.

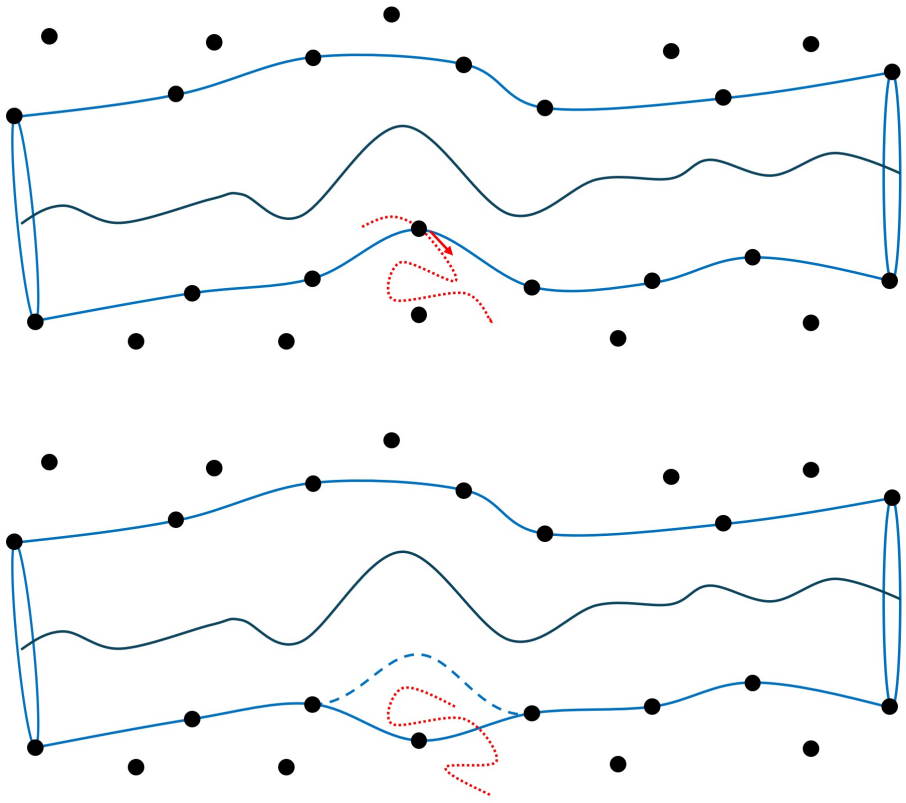


Figure 2.4: A figure to show how a constraint release event can occur.

As the name suggests, relaxation due to constraint release is not relaxation

caused by movement of the chain, such as reptation or CLF, but rather due to a change in the constraining tube itself. Whenever the end of one of these surrounding chains passes through the edge of the constraining tube, a constraint release event occurs, and the constraints imposed by that segment of the tube are lost. These constraint release events can occur as the surrounding chains relax via reptation, known as reptative or thermal constraint release, or by convection as the surrounding polymer chains are swept away by the polymer flow, known as convective constraint release. The movement of these surrounding chains is affected by the rate of an imposed flow, and thus the rate of CCR events grows in proportion to the flow rate. This mechanism was used to explain the results that exhibited a relaxation rate that was faster than expected under the Doi-Edwards model.

Several models were then created to incorporate these effects of CCR. Many of the early models for polymer dynamics were contour-variable models. A contour-variable model keeps track these chain contours change over time, with models based on how these chains interact at a molecular level. The Doi-Edwards tube model is one such example. This formulation often naturally leads to systems of PDEs where the chain is split into segments and the response of each segment is dependent on the other segments. On the other hand, non-contour-variable models do not model at the dynamics of the chain at the molecular level, instead directly deriving an ODE for the stress response. A number of early models derive their ODEs by taking an average along the chain. Contour-variable models tend to be more accurate but also more expensive than non-contour-variable models.

The Mead-Larson-Doi (MLD) model [25] inherits the structure of the Doi-Edwards model, replacing the term of reptation with a combined relaxation term that encapsulates the effects of both reptation and CCR. This

contour-variable PDE is then reduced to a non-contour-variable ODE via overall chain considerations. Like the Doi-Edwards model before it, the MLD model solves for the full chain distribution, with both the accuracy and computational cost associated with modelling at the molecular level. However, reptation and CCR act differently on a polymer chain at the molecular level. Relaxation by reptation acts on the ends of the chains, and thus the ends relax much faster than the centre under reptation as the effects take a long time to diffuse along the chain. On the other hand, CCR is often envisioned as allowing Rouse-like motion of the tube as a whole, without a focus on a particular area of the chain. The MLD model does not capture these mechanisms which are widely regarded as correct, this fidelity is lost when the MLD model groups them together in a single term.

Many other early approaches postulated the ODE directly with non-contour-variable models such as the model by Ianniruberto and Marrucci [17]. They directly used the proposed physics of CCR to inspire their model but did not directly derive the ODE from the molecular level. Thus this ODE model was a lot easier to use with faster speeds than a full solution, but it lacked the robustness that an equivalent ODE derived from molecular principles would have. Many of these models had greater experimental agreement than their predecessors and were crucial to furthering understanding in the field of polymer dynamics, but it was not until the creation of the GLaMM model that a model would be able to accurately capture a wide range of different rheological features.

2.3 The GLaMM model

Building upon the Doi-Edwards tube model and the Likhtman-McLeish theory [22], the GLaMM model is a contour-variable non-linear PDE for the prediction of polymer dynamics created in 2003 [14]. Unlike most other polymer models of the time, the GLaMM model was based on a stochastic model for chain motion, derived from the Brownian motion of the polymer chains themselves. Such an approach followed naturally when starting from the molecular physics of polymers, and allowed for the implementation of all the physics at the length scale of the tube length. This had a number of advantages. One major advantage was that this direct derivation from physics provided a natural way to include CCR, as interactions between chains. This was what really set the GLaMM model apart from prior models, the fact that it could directly model CCR as Rouse motion of the tube. Another major advantage was that this model did not require parameter fitting to capture nonlinear rheology, unlike previous models. The only parameters that required fitting for the GLaMM model derivation were the ones directly inherited from the Doi-Edwards tube model, which were fitted with linear rheology. Thus, the accuracy of nonlinear rheology predictions exhibited by the GLaMM model must be a predictive result from the GLaMM model's understanding of the underlying physics, rather than a result of parameter fitting to similar experiments. One weakness of the GLaMM model, which is typical for this level of modelling, is that the GLaMM model derivation requires a closure approximation based on pre-averaging assumptions. There are also certain advanced features which the GLaMM model fails to capture, especially in strong stretch, but overall the GLaMM model describes a significant portion of experimental data well with a small number of parameters.

As we use the GLaMM model PDE extensively in this thesis, we shall describe the relevant parts of the model here. The dependent variable for the GLaMM model is a 3 by 3 tensor field $\mathbf{f}(s, s', t)$, and thus we must first introduce this quantity in order to understand the GLaMM model. Let us denote a single polymer chain by $\mathbf{R}(s)$, where $s \in [0, Z]$ runs along the chain length in units of entanglement segments. One such example of $\mathbf{R}(s)$ is illustrated in figure 2.5.

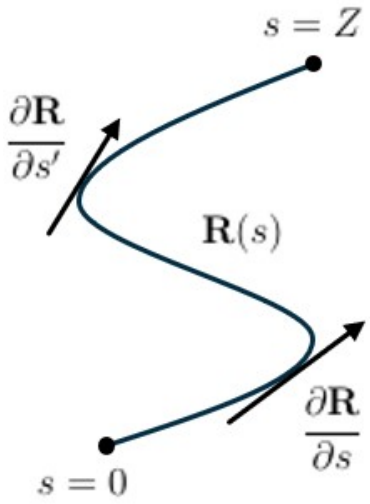


Figure 2.5: A visualisation of a polymer chain, $\mathbf{R}(s)$, with gradient vectors.

Let us pick two points on this chain at s and s', with gradient vectors $\frac{\partial}{\partial t}\mathbf{R}(s)$ and $\frac{\partial}{\partial t}\mathbf{R}(s')$. Using the components of these gradient vectors we can now define $\mathbf{f}(s, s')$ as,

$$f_{\alpha\beta}(s, s') = \Big\langle \frac{\partial R_\alpha(s)}{\partial s} \frac{\partial R_\beta(s')}{\partial s'} \Big\rangle \quad \text{where} \quad \mathbf{R}(s) = (R_x(s), R_y(s), R_z(s)) \tag{2.1}$$

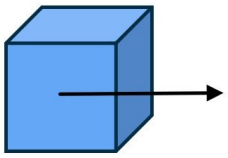
Where the angular brackets denote an ensemble average, calculated as the average as the number of simulated chains tend to infinity. α and β come from the components of the gradient vectors and $\alpha\beta$ will now be referred to as the Cartesian components of $\mathbf{f}(s, s')$. $\mathbf{R}(s)$ is based upon the stochastic process of a random walk and this means that we can use the ensemble

average to find an average function for $\mathbf{f}(s, s')$ as the number of $\mathbf{R}(s)$ we simulate tends to infinity. Further details into the statistical mechanics behind the creation of the GLaMM model are not necessary to understand the methods in this work and thus are omitted. We now break down $\mathbf{f}(s, s')$ for further clarity,

$$\mathbf{f}(s, s') = \begin{pmatrix} f_{xx}(s, s') & f_{xy}(s, s') & f_{xz}(s, s') \\ f_{xy}(s, s') & f_{yy}(s, s') & f_{yz}(s, s') \\ f_{xz}(s, s') & f_{yz}(s, s') & f_{zz}(s, s') \end{pmatrix}.$$

Using this, we can see that $\mathbf{f}(s, s')$ is a symmetric 3x3 tensor field, containing 6 distinct 2D functions of s and s'. As a polymer is subjected to a variation in conditions, the average path of a polymer chain will change, and thus so too will each of these 6 functions. The GLaMM model takes $\mathbf{f}(s, s')$ to be a time-dependent variable and presents a numerical solution to the evolution of this tensor field over time, $\mathbf{f}(s, s', t)$. To model the stain rate we apply to the polymer melt, we introduce a velocity gradient tensor $\boldsymbol{\kappa}$. Some examples of simple flows that polymers are regularly subjected to are shown in 2.6.

Shear $\quad \boldsymbol{\kappa} = \begin{pmatrix} 0 & \dot{\gamma} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$



Uniaxial extension $\quad \boldsymbol{\kappa} = \begin{pmatrix} \dot{\epsilon} & 0 & 0 \\ 0 & -\frac{\dot{\epsilon}}{2} & 0 \\ 0 & 0 & -\frac{\dot{\epsilon}}{2} \end{pmatrix}$

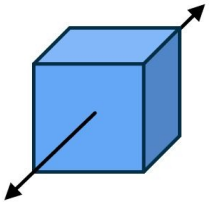


Figure 2.6: $\boldsymbol{\kappa}$ for shear at rate $\dot{\gamma}$ and uniaxial extension at rate $\dot{\epsilon}$.

When in the isotropic rest state, $\mathbf{f}$ takes the equilibrium value of $\mathbf{f}^{eq}$,

$$\mathbf{f} = \mathbf{f}^{eq} = \frac{\Delta(s-s')}{3}\mathbf{I} \tag{2.2}$$

Now, utilising these definitions, we can present the form of the GLaMM model we use throughout this thesis. The formulation we choose to use accounts for the effects of the reptation, CCR and retraction relaxation mechanisms. This form of the GLaMM model PDE can be written as,

$$\begin{aligned}\frac{\partial \mathbf{f}}{\partial t} &= \boldsymbol{\kappa}\cdot\mathbf{f} + \mathbf{f}\cdot\boldsymbol{\kappa}^T + \frac{1}{3\pi^2 Z\tau_e}\left(\frac{Z}{Z^*(t)}\right)^2\left(\frac{\partial}{\partial s}+\frac{\partial}{\partial s'}\right)^2\mathbf{f} \\ &+\frac{3\nu a}{2}\left[\frac{\partial}{\partial s}\left(\frac{a}{\sqrt{\mathrm{Tr}\,\mathbf{f}(s,s,t)}}\frac{\partial}{\partial s}(\mathbf{f}-\mathbf{f}^{eq})\right)+\frac{\partial}{\partial s'}\left(\frac{a}{\sqrt{\mathrm{Tr}\,\mathbf{f}(s',s',t)}}\frac{\partial}{\partial s'}(\mathbf{f}-\mathbf{f}^{eq})\right)\right] \\ &+\frac{R_s}{2\pi^2\tau_e}\left[\frac{\partial}{\partial s}\left(\mathbf{f}\frac{\partial}{\partial s}\ln[\mathrm{Tr}\,\mathbf{f}(s,s,t)]\right)+\frac{\partial}{\partial s'}\left(\mathbf{f}\frac{\partial}{\partial s'}\ln[\mathrm{Tr}\,\mathbf{f}(s',s',t)]\right)\right].\end{aligned} \tag{2.3}$$

Where $\cdot$ indicates a dot product across the Cartesian dimensions of the two operands, as is standard practice in rheology. In order, the right-hand side of equation (2.3) contains two terms to represent the effects of convection, and three terms with account for the relaxation due to reptation, CCR and retraction respectively. We will now define each of the other variables within equation (2.3) that have yet to be defined within this work. Firstly the variable a is the tube diameter for the Doi-Edwards tube model and R_s is a closure parameter taking the value of 2. ν is the CCR rate calculated as a product between the retraction rate, which is stretch-dependent and determines the rate at which CCR events occur, and a constant C_ν which describes the size of the effect on chain relaxation per event. We will use different values of C_ν throughout this thesis to simulate flows with and without the effects of CCR. $\tau_e = \tau_R/Z^2$ is the Rouse time of an entanglement segment. The function $Z^*(t)$ is known as the effective number of entanglement segments, accounting for how chain stretch changes the

contour length and thus affects the effective number of entanglement segments, the full formula can be found in the GLaMM paper [14]. Also note for readability, $\mathbf{f} = \mathbf{f}(s, s', t)$ unless otherwise specified. When calculating numerical solutions using the GLaMM model, the (s, s') space is discretised with an N by N grid, allowing this PDE model to be treated as a system of coupled ODEs. This discretisation imposes that $Z \in \mathbb{N}$ so that each entanglement segment, Z, contributes equally, and performs best when the mesh density $N \in \mathbb{N}$ is an odd multiple of Z.

The reason knowledge of how $\mathbf{f}(s, s', t)$ evolves is important is because this molecular scale quantity can be used in various calculations to derive many quantities important to the dynamics of the entire system. Total polymer stress is one such example,

$$\sigma_{\alpha\beta}(t) = \frac{12G_e}{5Z} \int_0^Z f_{\alpha\beta}(s, s, t) ds \tag{2.4}$$

Where G_e is a modulus constant that alters the resulting units. Although the GLaMM model PDE has a long computation time, the accuracy of the GLaMM model in predicting the behaviour of various polymer melts has been shown experimentally.

A final remark to note for this thesis is that $\mathbf{f}$ is a fast-evolving quantity and for an accurate numerical prediction, the time step we use must be approximately a tenth of the fastest time scale within the model or shorter. This requires $\delta t \leq (\tau_e Z^2/N^2)/10 = 1/90$ in the GLaMM model for time step convergence. Often it is not the absolute extension rate or shear rate that is measured, but the rate of the flow relative to the Rouse time of the material, $\dot{\epsilon}\tau_R$ or $\dot{\gamma}\tau_R$ respectively. This quantity is the Weissenberg number.

2.4 The Rolie-Poly model

The GLaMM model serves as a bridge between molecular simulations to an expensive but workable model, providing a way to retain a good amount of accuracy for polymer predictions without the cost of simulating each molecule. Despite this, there are still many situations where the GLaMM model is too expensive to be usable. For this reason, the derivation of the Rolie-Poly model [21] soon followed. The Rolie-Poly model similarly serves as a bridge, this time between the expensive contour-variable GLaMM model PDE and the much cheaper non-contour-variable Rolie-Poly ODE. It achieves this increase in computation speed by assuming all Fourier modes except the largest one are negligible, sacrificing molecular fidelity for computation speed. More details on Fourier modes and their relationship with fast polymer dynamics models are expanded upon in appendix A. The ease of use of the Rolie-Poly model led to widespread adoption in both polymer research and manufacture, despite many known situations where the Rolie-Poly model does not agree with experimental data as well as the GLaMM model [21]. This divergence from real-world results is the factor that inspired our aim. Not only would a model of similar speed but greater accuracy benefit direct predictions that currently use the Rolie-Poly ODE, but the also prediction of other features which use models based on the Rolie-Poly model, thus inheriting its flaws.

Unlike the GLaMM model which predicts the evolution of polymer chains, the Rolie-Poly model is an ODE for the stress tensor. We write the Rolie-Pole model as,

$$\frac{\mathrm{d}\boldsymbol{\sigma}}{\mathrm{d}t} = \boldsymbol{\kappa}\cdot\boldsymbol{\sigma} + \boldsymbol{\sigma}\cdot\boldsymbol{\kappa}^T - \frac{1}{\tau_{df}}(\boldsymbol{\sigma}-\mathbf{I}) - \frac{2(1-\sqrt{3/\mathrm{Tr}\boldsymbol{\sigma}})}{\tau_R}\left(\boldsymbol{\sigma} + \beta\left(\frac{\mathrm{Tr}\boldsymbol{\sigma}}{3}\right)^{\delta}(\boldsymbol{\sigma}-\mathbf{I})\right), \tag{2.5}$$

where,

$$\tau_{df}(Z) = \tau_d(Z)\left(1 - \frac{3.38}{\sqrt{Z}} + \frac{4.17}{Z} - \frac{1.55}{Z^{3/2}}\right).$$

It is worth noting the similarities in structure between the Rolie-Poly model and the GLaMM model. In equation (2.5) the right-hand side consists of two terms to model convection, a reptation term, and a term to model the effects of both CCR and retraction, with the part that has a factor of β accounting for CCR. This β is a constant that details the extent of relaxation per CCR event, similar to the role of C_ν in the GLaMM model, with $\beta = 10C_\nu$. This relation was derived empirically as the form of model reduction meant that a direct relation to the constant from the GLaMM model was not possible. Another parameter that we have yet to define is δ which determines how strongly the effects of CCR are suppressed by chain stretching. A value of -0.5 is established [21] to match the physics of the model and give results that are close to the GLaMM model. We also utilise an improved reptation rate, τ_{df}, devised by Likhtman and McLeish [22] to account for the fact the contour length fluctuations increase the speed at which the model relaxes by reptation.

Results from the Rolie-Poly model can be made dimensionless by multiplying by a factor of $4\tilde{G}_f/5$ where $\tilde{G}_f$ is given by the formula,

$$\tilde{G}_f(Z) = 1 - \frac{1.69}{\sqrt{Z}} + \frac{2}{Z} - \frac{1.24}{Z^{3/2}}.$$

These dimensionless values for stress can then be compared to results from the GLaMM model. The fastest time scale within the Rolie-Poly model is the Rouse time of the model, τ_R, which must be considered when choosing an appropriate time step for numerical predictions. Note here that we use the revised coefficients τ_{df} and $\tilde{G}_f$ suggested by Likhtman-McLeish theory [22] to give the Rolie-Poly model the best possible chance at pre-

dicting polymer behaviour, allowing us to make fair comparisons when we introduce our new model.

2.5 Other post-GLaMM models

A number of models have been created after the GLaMM model, with a significant portion of them aiming to improve modelling in areas where the GLaMM model is not as accurate such as under strong stretch. These studies identify some new physics that the GLaMM model does not currently include and then create a model with this new consideration. Examples include rate-dependent chain friction or tumbling. Broadly speaking there are three main approaches to identify these new physics and implement them into a new model.

Firstly there is the molecular dynamics driven approach. When simulating the interactions between molecules directly, it is possible to see molecular behaviours that are not yet considered by the GLaMM model. Direct visual evaluation of molecule movement can often give us greater insight into physical features that are very hard to discern from data alone. Tumbling is one such example that is identifiable in molecular dynamics simulations. Tumbling is the process by which a chain is initially aligned in the flow, some small Brownian motion occurs at the leading end of the chain, which the flow can then pick up, amplifying the movement of this edge, causing the chain to "tumble" out of alignment. The work by Sefiddashti et al. [31] compares molecular dynamics simulations of a polymer melt to various variations of the tube model. In this work, they identify the circumstances under which features these models often neglect occur, measure the size and describe the source of each divergence, and then suggest a modifica-

tion to account for some of these features. Another identifiable signature in molecular dynamics simulations is that of disentanglement under flow. CCR theory suggests that the constraints on any particular chain are lessened as the surrounding chains are also becoming aligned with the flow and that as the flow relaxes back to an isotropic rest state, these constraints return at the same rate. On the other hand, disentanglement under flow instead suggests that once these constraints are removed these disentangled chains take longer to recover their constraints than CCR implies. The work by Dolata et al. [9] adds certain entanglement dynamics, such as disentanglement under flow, to the Rolie-Poly model. They derive their models from thermodynamic principles and then compare their performance to prior models. Both these works are examples of how molecular dynamics simulations can be used to identify features and then use them to inform improved models that better describe polymer dynamics.

Secondly, there is the experiment-driven approach. This approach starts by finding some experimental data that the GLaMM does not accurately capture. Then measuring this difference under repeated experiments with slightly different conditions, and creating physical interpretations to explain the discrepancies. Rate-dependent chain friction is one example of a feature that was created to explain discrepancies with experiments. Taghipour et al. [37] analyse the agreement between the GLaMM model and experimental data, highlighting its strengths and limitations. They discuss possible revisions to the GLaMM model and inclusion of features such as chain tumbling before presenting an alternative approach that uses a steady-state regime that also includes their suggested revisions. The resulting model is able to reproduce steady-state viscosities accurately for a range of experimental data. The subsequent work by Hannecart et al. [15] explores the inability of previous tube-based models to predict scaling dependencies

that would correspond to experimental data and the proposition that this can be explained by a reduction in friction. As this feature is not easy to implement into the tube model, an alternative approach that considers the effects of friction reduction is proposed. They utilise experimental data to build upon this concept and reach a good agreement with experimental results after the inclusion of one further parameter.

Lastly, there is the non-equilibrium thermodynamics approach. Rather than simulations or experiments, this approach uses the inviolable rules of non-equilibrium thermodynamics as a basis to create new rheological models. By reformulating these rules in the context of polymers, they can guide model derivation or, conversely, suggest modifications that could be made to existing models that would make them abide by these laws. By ensuring that such laws are not broken, the physics of the problem can be preserved, often providing insight into potential improvements during the process of ensuring such laws are conserved. The work by Stepanou et al. [36] presents a way in which parameter values may be modified and new terms may be added to make an existing rheological constitutive equation thermodynamically consistent. This includes a method to approximate the effects of molecular tumbling whilst remaining thermodynamically admissible. The final modified model shows good agreement for its simplicity, making it viable in more expensive numerical calculations. The aforementioned work of Dolata et al. [9] also uses the requirement of thermodynamic consistency as a tool to inform their derivation.

Using these works we can get an idea of the current landscape for research around tube-based models, and the prevalent direction is to identify new polymer physics and then implement it into a new or existing model. Whilst there have been many successes in this field, we shall not take this route and add emergent features to the GLaMM or Rolie-Poly model. Whilst there

are some features the GLaMM model cannot capture, it remains a very good predictor for polymer dynamics with its most significant limitation being its computational cost. This fact is supported by widespread usage of the Rolie-Poly model, a reduction of the GLaMM model with a focus on speed over accuracy. Thus, this work shall instead investigate the lesser explored avenue of reducing the GLaMM model for a model that combines accuracy and speed. In addition, by creating a generalised framework for such an approach, physics introduced at the same molecular scale as the physics behind the GLaMM model can use the same pipeline to produce updated reduced models.

2.6 Applications of a reduced model

There are a number of applications that would significantly benefit from a more accurate model reduction of the GLaMM model, one of which is the modelling of polydisperse polymers. The previous models detailed in this literature review are all models for predicting the dynamics of monodisperse polymers. A monodisperse polymer material is composed of polymer chains which all have one uniform length. Whilst this type of material is much easier to model, monodisperse materials are rare even in laboratory experiments. Almost all applications of polymers use polydisperse polymer materials, which consist of a range of different chain lengths with frequencies detailed in an accompanying molecular weight distribution. The dynamical response of a polymer is heavily dependant on chain length, Z, as this directly affects the rate at which its various methods of relaxation occur, especially when two chains of different lengths interact. The most relevant model to discuss in this thesis is the Rolie-Double-Poly model [6] due to its connection to both the GLaMM and Rolie-Poly models that

define our aim. By discretising the molecular weight distribution into n distinct weights, this methodology uses the Rolie-Poly model to evaluate the interaction between each possible pairing of chain lengths, from which it can determine the dynamics. Other works such as the one by Mead et al. [26] take their own approach to the creation of a low-cost polydisperse constitutive model, and a number of them, Mead et al. included, similarly use this approach where each interaction between every possible pairing is evaluated. For the Rolie-Double-Poly model, there are many situations in which the model predictions depart from the experimental results due to the inherited inaccuracies from the Rolie-Poly model. Despite this, it remains widely used as it is one of the few low-cost polymer models applicable to polydisperse materials. The GLaMM model was deemed unsuitable for this framework due to its long computation time. We aim to create a model that is sufficiently cheap such that it can be used in the Rolie-Double-Poly framework to allow for the creation of a new polydisperse model in future.

Another application is the prediction of shear banding effects in polymers. One of the simplest ways to set up a shear flow in a liquid for examination is to create a channel of fluid between two walls, one stationary and one with some constant velocity parallel to the channel. This imposes a shear flow on the liquid. If we were to fill the channel with a Newtonian liquid, we would have a uniform shear flow across the width of the channel. For a time it was also believed that polymers experienced a uniform shear flow, but further research has shown that this is not necessarily the case. Instead different shear "bands" can form, with a low-shear band near the stationary wall, and a high-shear band against the moving surface. The effect is called shear banding and works such as Adams et al. [1] have modelled this behaviour with existing polymer models. This paper performs linear stability analysis on a form of the Rolie-Poly model and shows how pertur-

bations trigger instabilities which in turn result in transient shear banding. Whilst it would be theoretically possible to use the GLaMM model for flows that exhibit shear banding, by its nature of having multiple bands, a flow that experiences shear banding is no longer uniform. A non-uniform flow then requires the creation of a flow field and the use of computational fluid dynamics to solve, where multiple evaluations per timestep mean that the Rolie-Poly model is often used in favour of the GLaMM model. Furthermore, the effects of polydispersity are known to have a significant effect on shear banding, further limiting the number of viable polymer models. If our model is suitable for use in polydispersity, it could be useful for numerical computational fluid dynamics with shear banding. However, low computational cost is not the only requirement for a model to be suitable for linear stability analysis, models that are cheap but sufficiently complex are not viable for linear stability analysis. Whilst linear stability analysis is likely to be difficult with our proposed framework, the improvement of numerical predictions alone would be a step toward furthering the understanding of shear banding.

A further application that would benefit from polymer modelling is 3D printing. A common type of 3D printing is "fused-filament-fabrication" (FFF) and works such as the one by McIlroy et al. [24] show evidence to suggest that polymer rheology has a significant effect on the strength of the finished product. FFF prints the desired object layer-by-layer and the strength of an object fabricated with such a method is determined by the entanglement between layers. During printing, each layer must be deposited relatively quickly as the strength of the weld between the previous layer and the new one from the nozzle depends on the interdiffusion and entanglement across the interface but the diffusion process slows down as the previous printed layer cools. However, as a polymer is quickly extruded

from a nozzle to be deposited on a surface for 3D printing, it is subjected to high stresses which disentangle the incoming melt, also reducing diffusion across the interface. McIlroy et al. model the various processes that occur across such an interface to determine the effects polymer rheology has on the final weld strength. As this modelling is highly dependant on entanglements between polymers, a reduced model that is sufficient to meet our aims would provide an improved prediction than estimations with the Rolie-Poly model.

A similar challenge during polymer manufacturing is that of extrudate swell. When a polymer melt is compressed through a die to shape the material, it will be subjected to an increased flow rate. If the polymer is subjected to this increase rate for a sufficient length of time, the polymer chains will disentangle. When the polymer then exits the other side of the die, the polymer will relax back to the isotropic rest state causing the polymer to expand or "swell" as it regains entanglements. The work by Robertson et al. [35] uses a modified Rolie-Poly model to predict extrudate swell over a range of realistic processes. This includes the effects of polydispersity. As this paper uses the Rolie-Poly model it could again benefit from a model that meets our aim to provide increased accuracy at a similar cost.

One final area that could benefit from our reduced model is that of flow-induced crystallisation (FiC). Understanding the rate of crystallisation and how it is affected by flow is essential to transition from a polymer melt to a finished solid polymer product when crystallisation is involved. Read et al. [34] created the polySTRAND model for computational modelling of FiC effects from a molecular approach. The polySTRAND model is applicable to a range of processing parameters and polydisperse polymers. For a model to be suitable for the prediction of FiC, it is also required to be

suitable for polydisperse materials as both FiC experimental data and the industrial processes that would benefit from FiC prediction use polydisperse polymers. An article by Graham [12] explores why FiC is simultaneously of great importance to industry yet very difficult to simulate cheaply. This difficulty comes from the fact that an accurate prediction of FiC requires the combination of many effects at different length scales, necessitating multiscale models combining different levels of coarse-graining. If our reduced model is sufficiently cheap to be used for polydisperse polymers, it may also be able to be used as another link in the multiscale modelling chain at a different length scale to the molecular approaches.

2.7 Our model reduction approach

Our aim of using a set of coarse-grained variables to predict a more complex model is a form of model reduction. The term model reduction refers to the creation of a simplified model for a dynamic system, losing predictive accuracy for a simpler model that has a faster calculation time or is suitable for use within some methodology that the original was not. The largest difference is that whilst many papers on model reduction focus on theoretical problems [4, 11], this work is for a real-world problem which will allow us to take a physically motivated approach. We could choose any set of quantities to be our coarse-grained variables but intuition about the field of polymer dynamics tells us that a quantity important to the overall dynamics of a polymer such as the stress is more likely to have similar molecular behaviour for similar dynamic response. In other words, these quantities are more likely to characterise $\mathbf{f}(s, s')$ configurations well and so may be a good place to begin. Due to this added guidance from the model being grounded in physics, our approach does not use any of

the more theoretical model reduction methods and so we shall not explore them in detail here, but we do acknowledge the parallels in overall intent to model reduction and so some of the ideas presented here may prove useful in that field.

2.8 Existing data-driven approaches in rheology

Our methodology instead aims to perform this model reduction with a data-driven approach. This is an emerging field within rheology brought about by the continued growth of available computational power. Various machine learning techniques such as Gaussian process regression and neural networks are employed to learn model parameters and the models themselves. Machine learning is often employed to determine parameters that are otherwise hard to access. Examples in rheology include finding coefficients for newly derived models [30], the recovery of model parameters from data [20] and improving existing parametrisations [10]. Other than predicting clearly defined physical quantities, machine learning can also be used to discern material composition from experimental data [19]. Machine learning is also being used in the creation and development of constitutive equations, with neural networks learning from synthetic data produced by other models [40, 18], and some approaches using knowledge of rheology to further inform their choice [7, 23]. One particular example, dissipative particle dynamics, uses machine learning to reduce the required number of simulations [41] and find new constitutive equations via coarse-graining [29]. Of particular relevance to this work are the recent publications by Howard et al. [16] and Weeratunge et al. [38]. Howard et al. use machine learning

to learn the particle stresses in suspension flows for situations where direct measurement is not feasible. This work provides an example of machine learning's ability to extract macroscopic variables when learning from data at the molecular level. Weeratunge et al. employ machine learning to determine optimised model parameters for multiple coarse-grained polymer simulations. This approach significantly accelerates the determination of model parameters, allowing quick access to different levels of molecular coarse-graining. In contrast, this work uses a data-driven approach to map from the coarse-grained variables to the full model. This is unique as we use both synthetic data and an underlying model to perform model reduction on the GLaMM model rather than create an entirely new model. A model that is created in line with our aim would have the advantage of retaining the microscale mechanics of the GLaMM model as we directly call the full model, whilst leveraging the slow-moving property of the coarse-grained variables to take much larger timesteps. It would also be possible to utilise machine learning for coarse-grained variable selection.

2.9 Aims of this thesis

Our aim in this thesis is to create a new model for polymer dynamics, one that is of comparable speed to the Rolie-Poly model but with accuracy similar to the GLaMM model by a new model reduction of the GLaMM model. Our decision to pursue such an objective came after we identified a number of applications that would directly benefit from the creation of such a model. One of the applications that a new model can hope to develop is the prediction of polydisperse polymers. A model for polydispersity requires an accurate but computationally cheap constitutive model, and so this became the foundation for our aim. One such model, the Rolie-Double-Poly,

is a model for polydispersity that utilises the formulation of the Rolie-Poly model. The main reason that this model does not use the GLaMM model is not that it would not be possible, but that the long computational time would render a polydisperse GLaMM model too impractical for use. Currently, many of the limitations of the Rolie-Double-Poly can be traced back to equivalent problems within the Rolie-Poly model. However, if another suitable model reduction for the GLaMM model that was of similar speed to the Rolie-Poly model existed, such as one that would satisfy our aim, it would likely be able to use the same framework to adapt it into a polydisperse model. Polydispersity is not the only area that is currently limited the use of the Rolie-Poly model, computational fluid dynamics would also benefit from a more accurate model of similar speed. The applications identified in section 2.6 are all areas that require polydispersity, computational fluid dynamics or both and would furthered by the development of new cheap but accurate polymer models.

We have discussed the reasoning for our objective, but not yet the details around our choice of approach. We have reviewed a number of examples of models that modify the GLaMM model in section 2.5 but these works, as with most others attempt to add to or replace the GLaMM model, so that it can capture a certain feature. Very few works are looking to provide new model reductions for the GLaMM model, and with the recent application of data-driven methods to rheological problems, performing data-driven model reduction on the GLaMM model appeared promising. For these reasons, we chose to create our new model by model reduction of the GLaMM model.

Chapter 3

Reduced model with uniaxial extension flows

The first step to achieving our aim is the creation of a framework for data-driven model reduction. Discoveries in our prior work [27] led us to choose such a formulation due to the problems encountered during that work's approach to model reduction. This prior work attempted to create a new model reduction to the GLaMM model using its Rouse modes as coarse-grained variables, and the initial stages of this research investigated the possibility of furthering this approach. However, early in this research, a counter example showed that such an approach would be unsuitable for GLaMM model prediction to the degree of accuracy we are looking for. More details can be found in appendix A. These results suggested that a Rouse mode based model reduction would not be a meaningful improvement on the Rolie-Poly model without the inclusion of additional Rouse modes in the reduced model. Whilst such a generalisation is certainly possible, we instead choose to create a more sophisticated model reduction framework that does not restrict itself to Rouse modes when choosing

coarse-grained variables.

Section 3.1 establishes this framework for the general case, and then the following sections 3.2-3.4 apply this data-driven model reduction to the GLaMM model and evaluate its accuracy in predicting a variety of uniaxial extension flows. Section 3.5 discusses the various elements to consider when choosing the coarse-grained variables for this framework and section 3.6 is an overview of the performance of our reduced model for prediction of polymer melts under uniaxial extension flows. We use uniaxial extension flows in this chapter to prove the reduced model's effectiveness for a simple example before extending to more complex flow configurations.

3.1 Our model reduction framework

3.1.1 Establishing the general framework

We will now outline our method for deriving a reduced model from a small set of slow-moving degrees of freedom. We will present the framework briefly in generality in this section, discuss the overall approach, and then section 3.2 provides a more detailed example by applying this framework to the GLaMM model to create a reduced model for the prediction of polymer melt dynamics.

Our approach addresses PDEs of the form

$$\frac{\partial \mathbf{f}}{\partial t} = \mathcal{G}[\boldsymbol{\kappa}(t), \mathbf{f}(\mathbf{s}, t)], \tag{3.1}$$

where $\boldsymbol{\kappa}$ is an external driving term, $\mathbf{s}$ is a vector of spatial arguments, $\mathbf{f} = \mathbf{f}(\mathbf{s}, t)$ is a tensor of the dependant variables we want to evolve, known

as the model configuration tensor, t is time and $\mathcal{G}$ is a functional that defines the PDE.

We now suppose that we have chosen some reduced variables $\mathbf{\Gamma}$, which depend on $\mathbf{f}(\mathbf{s})$ via the equations

$$\begin{aligned} \Gamma_1 &= h_1[\mathbf{f}(\mathbf{s})] \\ \Gamma_2 &= h_2[\mathbf{f}(\mathbf{s})] \\ &..... \\ \Gamma_k &= h_k[\mathbf{f}(\mathbf{s})]. \end{aligned} \tag{3.2}$$

With this formulation, each Γ_i is one of our coarse-grained variables. We now assume that a given value of $\mathbf{\Gamma}$ specifies cleanly the overall configuration $\mathbf{f}(\mathbf{s})$. In other words, though there may be many ways to arrive at a specific $\mathbf{\Gamma}$, these different histories do not produce appreciably different configurations $\mathbf{f}(\mathbf{s})$. Whilst this assumption holds true, the mapping in equation (3.2) is almost invertible. The choice of the coarse-grained variables $\mathbf{\Gamma}$ and how well this obeys the above assumption is key to the accuracy of the reduced model.

If $\mathbf{\Gamma}$ specifies $\mathbf{f}(\mathbf{s})$, then we can derive a reduced model as follows:

1. Determine $\mathbf{f}(\mathbf{s})$ from $\mathbf{\Gamma}$.
2. Determine the dynamics of $\mathbf{\Gamma}$ from $\mathbf{f}(\mathbf{s})$ and the original PDE.

For step 1 we use a data-driven approach. We begin with a training set of sample transient solutions to equation (3.1). We suppose these use a range of choices of $\boldsymbol{\kappa}(t)$ so that the space of $\mathbf{\Gamma}$ is well covered. From this training set, we can map any value of $\mathbf{\Gamma}$ to its corresponding $\mathbf{f}(\mathbf{s})$ by interpolation of the training set. For step 2, substituting the interpolated $\mathbf{f}$ into equation

(3.1) provides $\frac{\partial \mathbf{f}}{\partial t}$. Combining $\frac{\partial \mathbf{f}}{\partial t}$ with the chain rule applied to equation (3.2) gives $\frac{\mathrm{d}\mathbf{\Gamma}}{\mathrm{d}t}$. Thus we have the dynamics of $\mathbf{\Gamma}$, which defines a reduced model from which we can obtain the dynamics of $\mathbf{f}(\mathbf{s})$.

3.1.2 Analysis of the framework in generality

Many constitutive equations can be expressed in the form presented in equation (3.1) from simple PDEs like the heat equation to complex mechanical or rheological models. The numerical solutions of some constitutive PDEs require a high temporal resolution in order to capture the finer details of the configuration of $\mathbf{f}(\mathbf{s})$. The major advantage of this method is that if we choose a suitable $\mathbf{\Gamma}$ that evolves slowly, we can bypass this requirement. With our approximation, we can reconstruct $\mathbf{f}(\mathbf{s})$ at any point from $\mathbf{\Gamma}$. When we then calculate the dynamics of $\mathbf{\Gamma}$ we use much larger timesteps than would have been possible with direct calculation of the PDE.

A good choice of $\mathbf{\Gamma}$ is one that upholds our framework's assumption whilst evolving slowly in time. In this work we have directly verified these qualities for our choices through direct testing, but it may be possible to automate this process by defining a metric that evaluates the accuracy of $\mathbf{f}(\mathbf{s})$ predictions for a given choice of $\mathbf{\Gamma}$. This can be seen in as part of the future work in chapter 5.2.

3.2 Application to the GLaMM model

In this section we explore the methodology in more detail, using the GLaMM model of polymer dynamics as an example. Utilising the general framework

we established in section 3.1.1, we now present how to apply it to a specific example, the GLaMM model for polymer dynamics [14]. The GLaMM model has the general form,

$$\frac{\partial \mathbf{f}}{\partial t} = \mathbf{f}(\mathbf{s},t)\cdot\boldsymbol{\kappa}(t) + \boldsymbol{\kappa}^T(t)\cdot\mathbf{f}(\mathbf{s},t) - \mathcal{H}[\mathbf{f}(\mathbf{s},t)]. \tag{3.3}$$

Written in this form, the first two terms are the standard upper-convected Maxwell derivative, used in many areas of rheology to model response to an imposed flow, and $\mathcal{H}$ is a collection of terms which describe the various polymer relaxation processes. In this specific case of equation (3.1), $\mathbf{f}(\mathbf{s},t)$ is a 3 by 3 tensor of functions dependant on $\mathbf{s}$ and t, where $\mathbf{s}$ is a vector of two continuous variables (s,s'). The external driving term, $\boldsymbol{\kappa}(t)$, refers to a flow imposed upon a polymer melt in the GLaMM model. Throughout this section we shall only use flows with uniaxial extension in the x direction, $\kappa_{xx} = \dot{\epsilon}, \kappa_{yy} = \kappa_{zz} = -\dot{\epsilon}/2, \kappa_{\alpha\beta} = 0 \;\forall\; \alpha \neq \beta$. Contributions from CCR are excluded for simplicity and because CCR has little effect in extension, hence $C_\nu = 0$.

We can also specify the definition of the set of h_i from equation (3.2). For the GLaMM model, we allow h_i to be any general functional of $\mathbf{f}(s,s')$. We present two such examples for choices of h_i for the GLaMM model under uniaxial extension flow here. We define

$$\Gamma_1 = \frac{\sigma_{xx}-\sigma_{yy}}{Z} = \frac{12}{5Z^2}\int_0^Z f_{xx}(s,s) - f_{yy}(s,s)\; ds \tag{3.4}$$

as the dimensionless first normal stress difference. For Γ_2 we take a similar integral on the off-diagonal as follows,

$$\Gamma_2 = \frac{12}{5Z^2}\int_0^Z f_{xx}(s,Z-s) - f_{yy}(s,Z-s)\; ds. \tag{3.5}$$

Γ_2 was created as a measure of deviation from affine deformation, to complement the abilities of Γ_1. Note the lack of time dependence in these formulations, that is intentional as we will use these functionals to evaluate many configurations of $\mathbf{f}(s, s')$ during our model, including ones that are not part of the current flow evolution $\mathbf{f}(s, s', t)$. More details on its conception and physical meaning can be found in section 3.5.1. Computationally, we evaluate $\mathbf{f}(s, s')$ at N^2 distinct points on a grid over s and s', as is standard for numerical evaluations of the GLaMM model, and so these integrals are correspondingly approximated by sums with N terms.

Formalising the assumption established in 3.1.1, for this time-dependent PDE system for $\mathbf{f}(s, s', t)$, we introduce a mapping between $\mathbf{f} \in \mathrm{F}$ and a vector of coarse-grained variables $\mathbf{\Gamma} \in \mathbb{R}^k$ along with its corresponding inverse. We define $g : \mathbb{R}^k \to \mathrm{F}$ and $h_i : \mathrm{F} \to \mathbb{R}$ respectively as,

$$\mathbf{f}(s, s') = g(\Gamma_1, \Gamma_2, \ldots \Gamma_k) \quad \text{and} \quad \Gamma_i = h_i(\mathbf{f}(s, s')) \quad \text{where} \quad i = 1, 2, \ldots k. \tag{3.6}$$

In this case, F is a vector space of functions of two variables, s and s', that return 3 by 3 matrix objects. The functionals h_i, such as the examples in equations (3.4) and (3.5), are applied directly to $\mathbf{f}$. It follows that the forward mapping from $\mathbf{f}$ to $\mathbf{\Gamma}$ is computable explicitly and is therefore without error. However, our assumption is that there also exists a backward mapping, g, from $\mathbf{\Gamma}$ to $\mathbf{f}$. The calculation of $\mathbf{f}$ from $\mathbf{\Gamma}$ is not explicitly known and is susceptible to errors if the mapping is poorly defined as a result of an unsuitable choice of $\mathbf{\Gamma}$. The size of these errors directly depends on the suitability of the current $\mathbf{\Gamma}$ choice to uphold our assumption, and a detailed breakdown of this prediction error can be found in chapter 3.5.

As proposed in section 3.1.1, mapping from $\mathbf{\Gamma}$ to $\mathbf{f}(s, s')$ uses a data-driven approach and is performed by interpolation on a pre-established training

set, with this training set chosen to give good coverage of the range of possible $\mathbf{\Gamma}$ values. For this section, the training data is comprised of uniaxial extension flows. We use transient solutions of the GLaMM model with a range of constant rates $\dot{\epsilon}\tau_R = \{0, 0.1, 0.2, 0.3, \ldots, 3\}$, and a single stepped rate solution where the flow initially has non-zero rate, $\dot{\epsilon}\tau_R = 2.1$, which is instantaneously switched to $\dot{\epsilon}\tau_R = 0$ at $t = 2320$. All these solutions use $Z = 25$ and $\tau_e = 1$ so that the time is in units of τ_e. For calculation of these solutions we choose to discretise the (s, s') space with a 75^2 point grid so that $N = 75$, and with this choice, a timestep of $\delta t = 1/90$ is required for convergence. We exclude data with very high strain as solutions from the GLaMM model are known to be unrealistic past a certain threshold. This training data shall be referred to as our two-functional training set, denoted by $\mathbf{f}^{\text{Train}}$ and Γ_i^{Train}.

We have established the form of the GLaMM model PDE in equation (3.3), chosen two functionals in equations (3.4) and (3.5), and created our two-functional training set. With this, we have all the necessary components to apply the methodology from 3.1.1 to the GLaMM model. Denoting variables that are calculated from our reduced model as $\mathbf{f}^{\text{Pred}}$ and Γ_i^{Pred}, the evolution of Γ_i^{Pred} is determined using the steps outlined below:

1. Determine $\mathbf{f}(s, s', t)$ from $\mathbf{\Gamma}(t)$ using $\mathbf{f}^{\text{Pred}} = g(\Gamma_i^{\text{Pred}})$, via an interpolator on the two-dimensional $\mathbf{\Gamma}$ space.

2. Determine the dynamics of $\mathbf{\Gamma}(t)$ from $\mathbf{f}(s, s', t)$ using the GLaMM model PDE, $\frac{\mathrm{d}}{\mathrm{d}t}\Gamma_i^{\text{Pred}} = \frac{\mathrm{d}h_i}{\mathrm{d}t}(\mathbf{f}^{\text{Pred}}, \frac{\partial}{\partial t}\mathbf{f}^{\text{Pred}})$ and then the Euler method in succession.

To clarify the distinction between $\mathbf{f}^{\text{Train}}$ and $\mathbf{f}^{\text{Pred}}$ further, $\mathbf{f}^{\text{Pred}}(s, s', t)$ is our reduced model's prediction of the flow evolution and at any point we can

apply the functionals h_i to find the corresponding $\Gamma_i^{\text{Pred}}(t)$. Our training data is a collection of vectors $\mathbf{\Gamma}^{\text{Train}}$ denoting points at which we have recorded the configuration $\mathbf{f}^{\text{Train}}(s, s')$. These training variables lack time dependence as this training data is not part of the current flow evolution.

For the first step, many methods are viable to interpolate these data, we choose to use an interpolator that we shall refer to as the quadrant interpolator. This interpolator interpolates from the nearest point in each of the four quadrants. This method is described in appendix B.1. For the second step, we can obtain $\frac{\partial}{\partial t}\mathbf{f}^{\text{Pred}}$ by applying the original PDE to this $\mathbf{f}^{\text{Pred}}$ from interpolation and then use the chain rule on equations (3.4) and (3.5) to determine $\frac{\mathrm{d}h_i}{\mathrm{d}t}(\mathbf{f}^{\text{Pred}}, \frac{\partial}{\partial t}\mathbf{f}^{\text{Pred}})$. With this choice of functional, we can write $\frac{\mathrm{d}}{\mathrm{d}t}\Gamma_i^{\text{Pred}}$ in terms of $\frac{\partial}{\partial t}\mathbf{f}^{\text{Pred}}$ alone as follows,

$$\frac{\mathrm{d}\Gamma_1}{\mathrm{d}t}(t) = \frac{12G_e}{5Z^2}\int_0^Z \frac{\partial f_{xx}}{\partial t}(s, s, t) - \frac{\partial f_{yy}}{\partial t}(s, s, t)\ ds, \tag{3.7}$$

$$\frac{\mathrm{d}\Gamma_2}{\mathrm{d}t}(t) = \frac{12G_e}{5Z^2}\int_0^Z \frac{\partial f_{xx}}{\partial t}(s, Z-s, t) - \frac{\partial f_{yy}}{\partial t}(s, Z-s, t)\ ds. \tag{3.8}$$

To complete the second step, we increment each Γ_i^{Pred} forward in time with these $\frac{\mathrm{d}}{\mathrm{d}t}\Gamma_i^{\text{Pred}}$ via the forward differencing Euler method. Following these steps, we are able to predict the evolution of the tube-diameter scale quantity $\mathbf{f}(s, s', t)$ whilst evaluating that evolution on the whole-molecule scale with the Euler method on $\mathbf{\Gamma}$. Provided we choose these Γ_i variables so that they evolve slowly as mentioned in our initial aims, this implies that $\mathbf{\Gamma}$ is not as sensitive to changes on the tube-diameter scale and thus we can utilise a much larger timestep.

If we wish to simulate a single flow from rest, then our initial condition for prediction with our reduced model is $\mathbf{f}^{\text{Pred}}(s, s', 0) = \mathbf{f}^{eq}(s, s')$ where $\mathbf{f}^{eq}$ is defined in equation (2.2). However, we can also delay the introduction of

our reduced model, calculating any number of timesteps with the GLaMM model and then introducing our model at delayed start time T with initial condition $\mathbf{f}^{\text{Pred}}(s, s', T) = \mathbf{f}^{\text{Test}}(s, s', T)$, where $\mathbf{f}^{\text{Test}}(s, s', T)$ is the value of $\mathbf{f}$ from the GLaMM model after time T has elapsed. Some later models in chapter 4 will introduce our reduced model after a single timestep delay to avoid singularities in some derivatives when $\mathbf{f} = \mathbf{f}^{eq}$.

Throughout this thesis we compare results from our reduced model with results from the GLaMM model under uniaxial extension flows and shear flows. We detail the common parameter values used in our GLaMM model calculations here. A previous figure, figure 2.6, shows how these two flows can be represented in the GLaMM model as a matrix $\boldsymbol{\kappa}$. Similarly to that image, $\dot{\epsilon}$ will denote the rate for a uniaxial extension flow, whilst shear flows will use $\dot{\gamma}$. Some earlier calculations shall exclude the effects of CCR with $C_\nu = 0$ whilst plots that include CCR shall use $C_\nu = 0.1$. We use $Z = 25$, $N = 75$ for the GLaMM calculation unless otherwise specified. Additionally, we work in units of space and time such that $a = 1$ and $\tau_e = 1$. This choice of variables leads to a Rouse time for our simulated polymer melt of $\tau_R = Z^2\tau_e = 625$.

3.3 Prediction with one functional

Our methodology in section 3.2 defines two functionals, so we shall first show that a reduced model with one functional is insufficient to predict the GLaMM model. To do so, this section will use our reduced model with the first normal stress difference from equation (3.4) as the only functional so that $\boldsymbol{\Gamma} = \Gamma_1$. A training set consisting of one transient solution of the GLaMM model is sufficient to span $\boldsymbol{\Gamma}$ space when only one functional

is used. Thus our one-functional training set is comprised of data from a single transient solution with $\dot{\epsilon}\tau_R = 3$. If Γ_1 alone can correctly capture the behaviour of $\mathbf{f}(s, s', t)$, we would see that the profiles of $\mathbf{f}(s, s', t)$ with the same Γ_1 value from different transient solutions are approximately equal, regardless of flow history. We test this by comparing a snapshot of a transient solution directly from the GLaMM model, $\mathbf{f}^{\text{Test}}$, to a prediction, $\mathbf{f}^{\text{Pred}}$, which uses our reduced model with the one-functional training data, in figure 3.1.

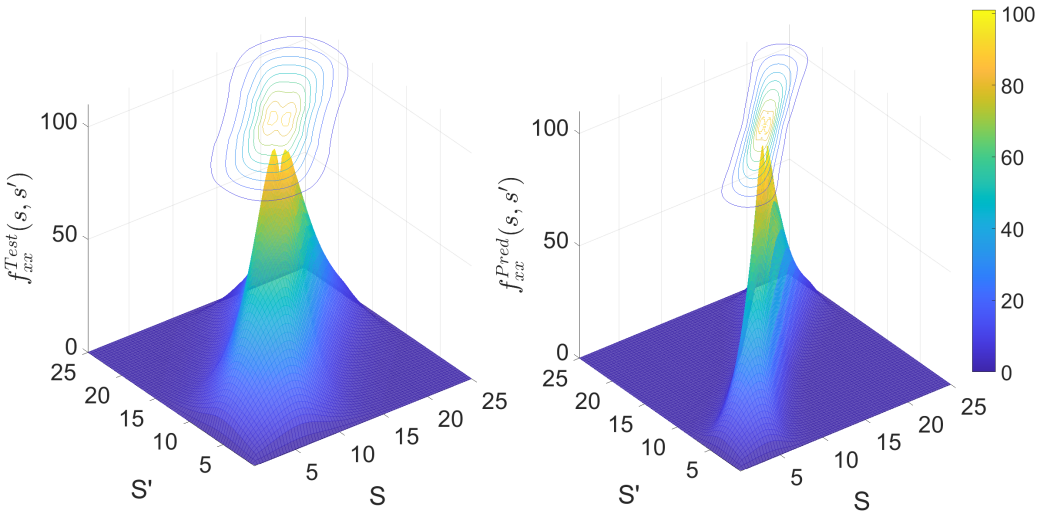


Figure 3.1: A plot showing the profiles for the xx, yy and xy components of $\mathbf{f}^{\text{Test}}$ and $\mathbf{f}^{\text{Pred}}$ with $\dot{\epsilon}\tau_R = 2.5$ and $C_\nu = 0$ at time = 1260, with $k = 1$ and Γ_1 from equation (3.4).

Figure 3.1 shows that knowledge of the value of the first normal stress difference is not sufficient to capture the width of $f_{xx}(s, s')$, even between two flows of similar rates. With only this Γ_1, the reduced model is unable to differentiate between the thin peak from the training data and the wider peak that it is asked to predict. It is not possible to definitively state that one functional is insufficient for any choice of Γ_1 without exhaustively testing all options, but some traits led us to believe that two functionals would be a more suitable route. Specifically, we tested multiple Γ_1 choices but in all cases, values of Γ_1 had multiple significantly different valid $\mathbf{f}$

configurations for each choice we tested. The $\mathbf{f}$ configurations appear to have multiple distinct features, suggesting that characterisation would need multiple distinct functionals. Another factor supporting the requirement of $k > 1$ is that the stress tensor, $\boldsymbol{\sigma}$, has two independent components under uniaxial extension, suggesting that at least two degrees of freedom in $\boldsymbol{\Gamma}$ space are required to fully capture this behaviour. These factors lend credence to the idea that two functionals would significantly improve prediction accuracy, and as our model is already defined in such a way that naturally extends to any number of functionals; this is the route we proceed with.

3.4 Prediction with two functionals

Figure 3.1 shows two $\mathbf{f}$ configurations that have the same first normal stress difference but the width of the peak is clearly different in the two cases. In this section we introduce a second functional to capture this difference in width and show that it gives a highly accurate reduced model. For this, we return to using the two functionals from equations (3.4) and (3.5) with quadrant interpolation on our two-functional training set from section 3.2. We discuss why this choice of Γ_2 is effective at capturing width in section 3.5.1. We begin by predicting the same $\mathbf{f}$ configuration from figure 3.1 with two functionals in figure 3.2, temporarily excluding the transient solution for $\dot{\epsilon}\tau_R = 2.5$ from the two-functional training set to prevent triviality.

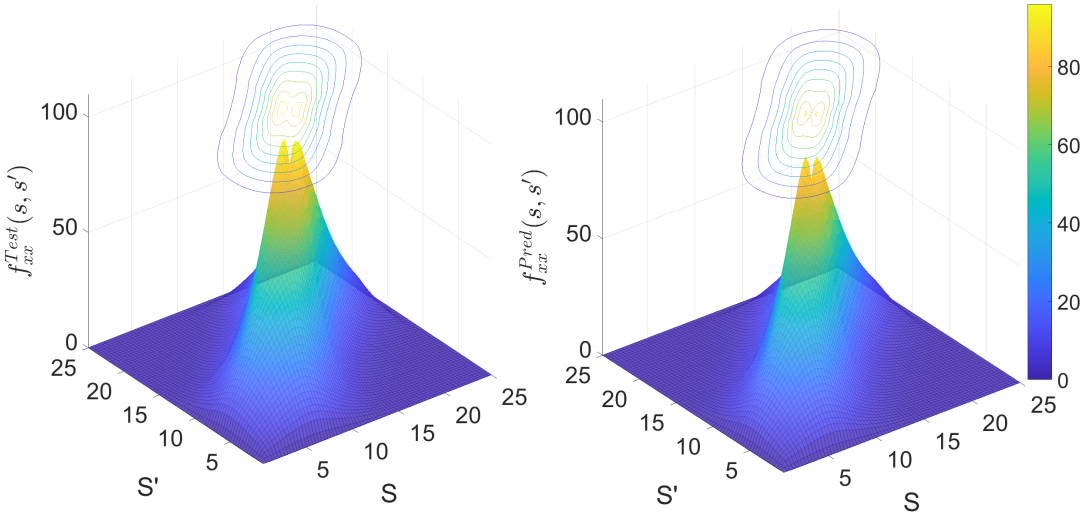


Figure 3.2: A plot showing the profiles for the xx, yy and xy components of $\mathbf{f}^{\text{Test}}$ and $\mathbf{f}^{\text{Pred}}$ with $\dot{\epsilon}\tau_R = 2.5$ and $C_\nu = 0$ at time $= 1260$, with $k = 2$ and Γ_1, Γ_2 defined in equations (3.4) and (3.5) respectively. For this figure, the transient solution for $\dot{\epsilon}\tau_R = 2.5$ was not included in the training set.

Figure 3.2 shows that the inclusion of a second functional significantly improves the accuracy of f_{xx}^{Pred} for this snapshot, implying an improvement in our reduced model's ability to distinguish between $\mathbf{f}$ configurations with different widths. With confirmation that we can predict one $f_{xx}(s, s')$ configuration well, we now use our reduced model to calculate the evolution of $\mathbf{f}^{\text{Pred}}$. Using our two functionals with our respective two-functional training set, we calculate the evolution of Γ_1 and Γ_2 for $\dot{\epsilon}\tau_R = 2.5$ in figure 3.3, once again, the $\dot{\epsilon}\tau_R = 2.5$ solution is excluded from the training set.

Figure 3.3 shows that our reduced model accurately predicts the evolution of both functionals at this flow rate, even at a timestep 900 times larger than the GLaMM model. Despite the high level of accuracy, before we can evaluate our assumption's accuracy we must first verify that the deviation between Γ_i^{Test} and Γ_i^{Pred} is due to our choice of functionals and that it is not subject to other sources of error.

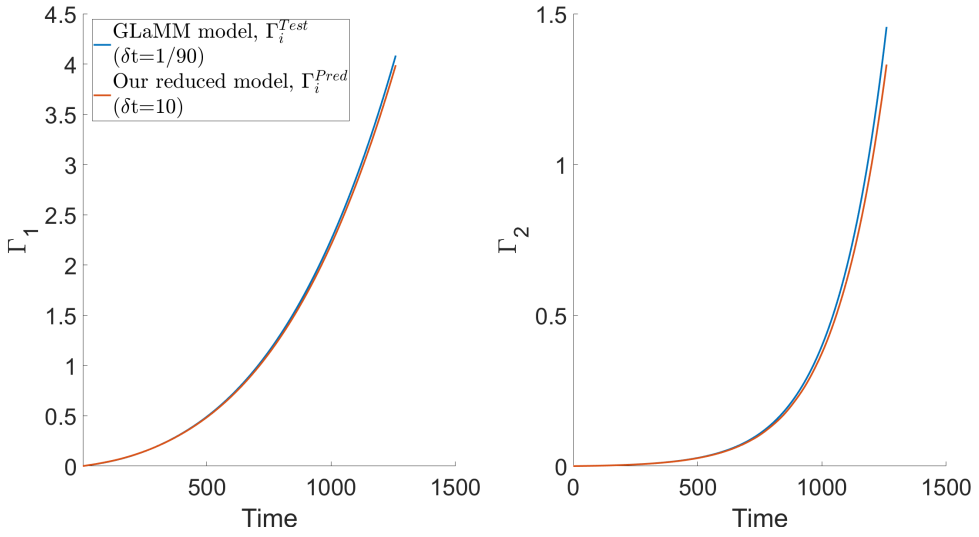


Figure 3.3: A plot comparing the evolution of Γ_i^{Test} and Γ_i^{Pred} for $\dot{\epsilon}\tau_R = 2.5$, $C_\nu = 0$. Γ_1 and Γ_2 are defined in equations (3.4) and (3.5). The two-functional training set is used, but the transient solution for $\dot{\epsilon}\tau_R = 2.5$ was not included.

3.4.1 Convergence errors

The disparity between Γ_i^{Test} and Γ_i^{Pred} in figure 3.3 may be due to errors in our prediction due to our choice of functionals, but there is also the possibility that these errors arise from our interpolation method or larger timestep size. When the training set is insufficiently dense, the interpolator may return inaccurate values for Γ_i^{Pred} as the prediction must use training points that are far away from the current location in $\mathbf{\Gamma}$ space. This can be alleviated most easily by increasing the number of GLaMM model solutions in the training set. However, the degree to which this error is affected by training point sparsity can be reduced with suitable interpolation methods, where an interpolator that is able to reduce this error to an acceptable level with fewer solutions in the training set is deemed to have higher suitability. The time step error is the result of numerical inaccuracies incurred through use of the Euler method in our formulation, causing an accumulation of errors over time. Time step error can be reduced by shortening the time step used in this calculation. Again, the time step required to reduce this

error to an acceptable level will depend on the functionals used. In theory we could add as much training data or shorten the time step as much as we like to reduce two other errors, but these will come at a computational cost, so we instead opt to find a balance between speed and accuracy.

We can also analyse the contribution to error from the interpolation method and time step size by predicting the GLaMM model with our new reduced model under different conditions and comparing the accuracy of the results. One such comparison can be seen in figure 3.4.

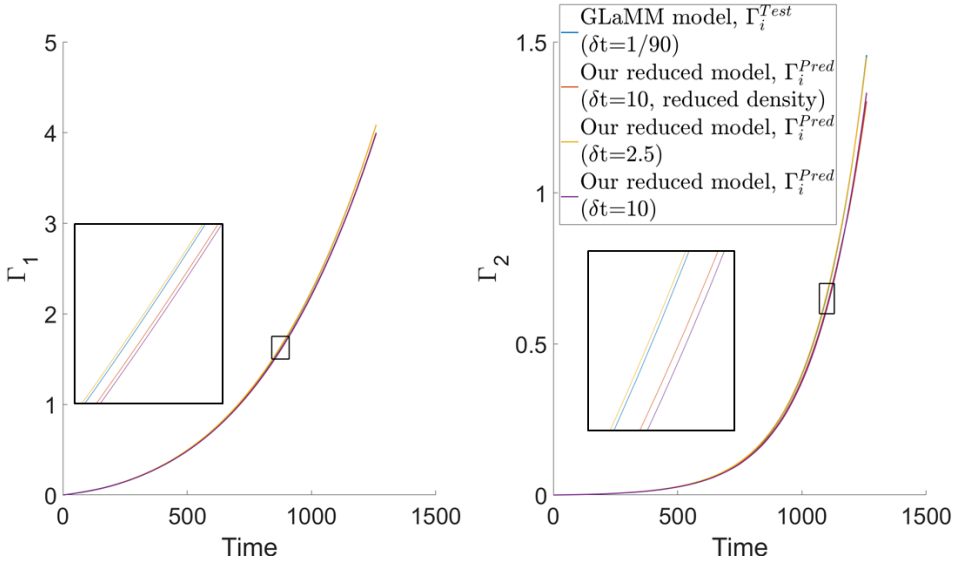


Figure 3.4: A plot comparing Γ_i^{Test} to Γ_i^{Pred} under various conditions. $\dot{\epsilon}\tau_R = 2.5$, $C_\nu = 0$. Γ_1 and Γ_2 are defined in equations (3.4) and (3.5).

Many details about the sources of error can be extracted from figure 3.4. Analysing two predictions with differing timesteps shows that whilst $\delta t = 2.5$ is a slight improvement, estimation using $\delta t = 10$ remains very accurate. With regards to interpolation error, the line that represents a prediction using a training set with a lower local density of training points around the test set has no significant difference in error. Further testing on different rates confirmed that $\delta t = 10$ was a suitable compromise between accuracy and speed, and that the two-functional training set was sufficiently dense so that the error due to the method of interpolation was negligible.

3.4.2 Constant rate extension

With the other sources of error accounted for, we can now test the veracity of our assumption without interference. We compare the accuracy between our new reduced model and another common model reduction of the GLaMM model, the Rolie-Poly model, at various rates, none of which are present in our two-functional training set.

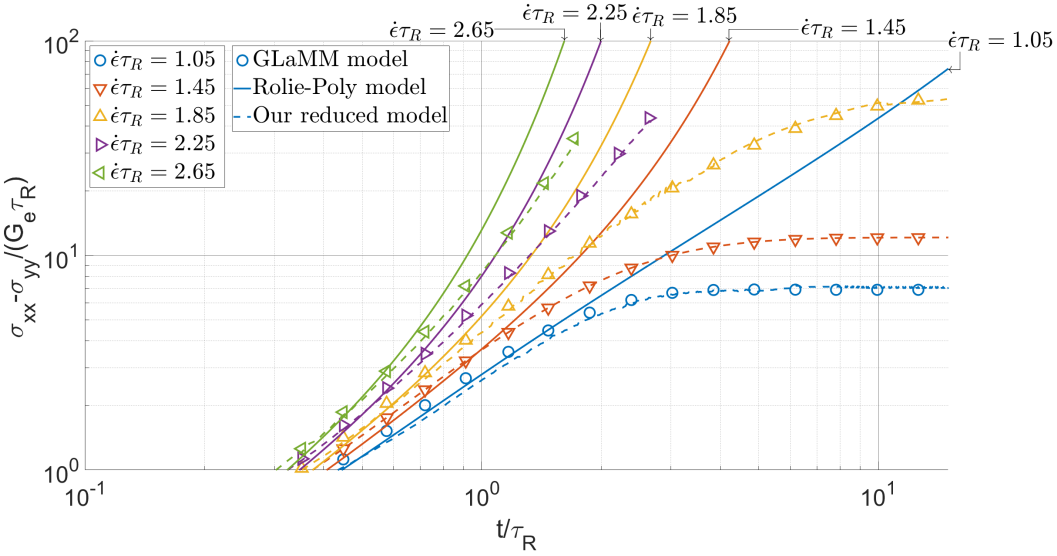


Figure 3.5: A figure to show the accuracy of first normal stress predictions using our reduced model under various different constant extension rates. The Rolie-Poly model predictions of the same rates are included for comparison.

In figure 3.5, both our reduced model and the Rolie-Poly model aim to capture the GLaMM model, and it is clear that our reduced model is significantly better. The reduced model also retains a large increase in computation speed compared to the GLaMM model, which can be attributed to two major factors:

1. The GLaMM model requires a time step every 1/90 units of t/τ_R to converge, whilst both the Rolie-Poly model and our reduced model retain negligible time step errors with a time step of 10 units of t/τ_R.

2. Utilising symmetries, if $\mathbf{f}(s, s', t)$ is discretised to a grid of $(N+1)^2$

points, the GLaMM method requires a minimum of $(N^2 - mod(N, 2))/4$ evaluations per time step, whilst our reduced method only requires $N - 1$ evaluations per timestep with the current choice of functionals.

For this data, we use a mesh density of $N = 75$ and thus, for each timestep, the GLaMM model evaluated the PDE over 10000 times as often as our reduced model. Performing GLaMM evaluations constitutes around 85% of the computation time for the reduced model and so it has a significant speed advantage over the GLaMM model. In summary, our reduced model is much faster than the GLaMM model for the computation of constant uniaxial extension flows whilst retaining accuracy.

3.4.3 Variable rate extension

Constant rate uniaxial extension flows are qualitatively similar to evaluations in our training set, so to test the robustness of our reduced model, we now predict a flow that is qualitatively different to the training data. The test data we chose to predict is a uniaxial extension flow of constant rate $\dot{\epsilon}\tau_R = 2.9$ until $t = 900$, at which time we instantaneously change the flow rate to $\dot{\epsilon}\tau_R = 0$. The training set and this test curve can be seen in figure 3.6.

This is why we included a single piecewise rate into the training set, resulting in our two-functional training set described in section 3.2. Without these extra data, inaccuracies would occur during the final stages of relaxation for some flow conditions. In its absence, we successfully predicted multiple non-constant rates for the majority of their evolution. However, the method predicted unrealistic behaviour when $\mathbf{f}$ was close to its equilibrium value $\mathbf{f}_{eq}$. After investigation, it was found that flows which had an instantaneous change to a lower rate would relax through a region of $\mathbf{\Gamma}$

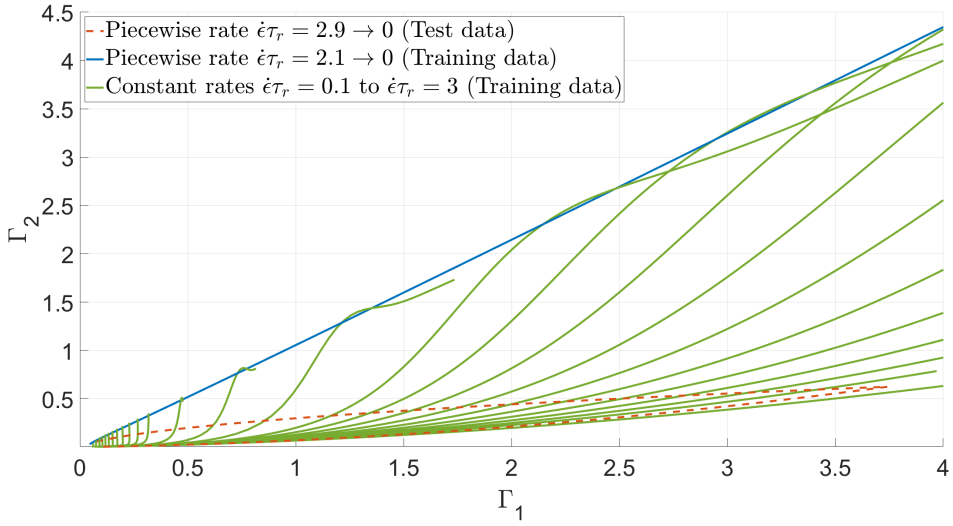


Figure 3.6: A plot showing the trajectories of the test and training data through $\mathbf{\Gamma}$ space.

space that was not covered by our training set of constant rate flows. In turn, this sparsity meant that our quadrant interpolator was unable to find a point in each of the four quadrants, suggesting that these predictions are less reliable. Thus we opted to add a single non-constant rate solution to the training set. This additional solution was subjected to an extension rate of $\dot{\epsilon}\tau_R = 2.1$ until $t = 2320$, at which point the flow was instantaneously stopped. The evolution of this single relaxation is able to sufficiently fill this empty region of $\mathbf{\Gamma}$ space in the training data so that relaxations to equilibrium are now captured with our new model for reduced rates. With this new rate, the reduced model is then used to generate figure 3.7.

Figure 3.7 shows that we are able to recreate the values of Γ_i for this test curve to a high degree of accuracy. Here we note that because the non-constant rate in the training set does not come into effect until very late times, we predict test data that exhibits qualitatively distinct behaviour to the solutions in our training set. The training set does include a single GLaMM solution of non-constant rate but this is only used by the interpo-

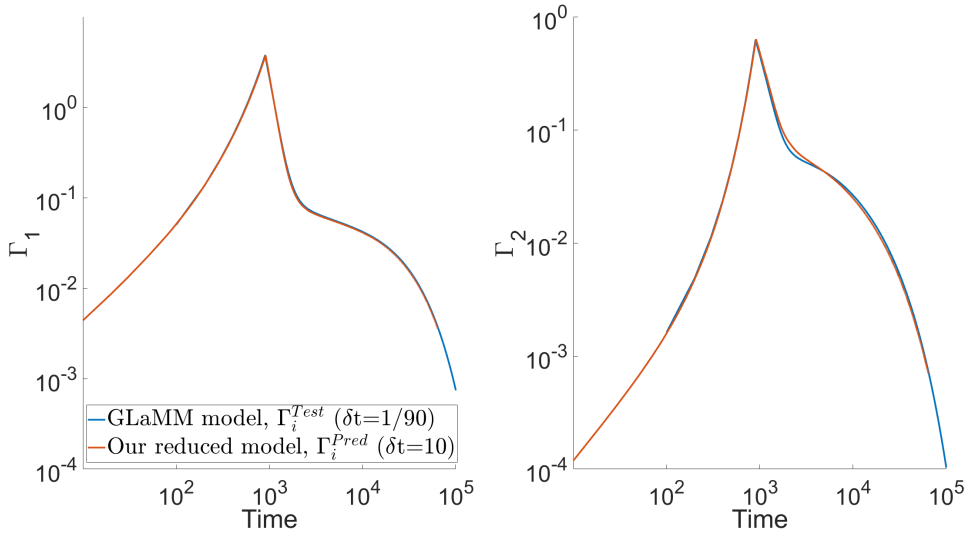


Figure 3.7: A plot comparing Γ_i^{Test} and Γ_i^{Pred} on a logarithmic scale for a flow of rate $\dot{\epsilon}\tau_R = 2.9$ for $0 \leq t < 900$ and $\dot{\epsilon}\tau_R = 0$ for $900 \leq t$, with $C_\nu = 0$.

lator for the final stages of relaxation back to the isotropic rest state, the sharp changes in gradient from the instantaneous changes in rate are all interpolated from constant rate extension GLaMM solutions alone. More specifically, when the flow is removed at $t = 900$ we are using training data that is under constant rate uniaxial extension flow to predict the test data that is under no flow and is purely relaxation. These are two fundamentally different flow histories and this indicates that we can predict various other flow histories, even in the absence of similar flow histories in the training set. With this result, we now compare GLaMM solutions to our reduced model and Rolie-Poly model results for flows that have instantaneous changes in rate in figures 3.8 and 3.9.

Figure 3.8 shows that our reduced model is able to correctly predict the stress evolution when the true solution contains qualitatively different behaviour to the solutions in the training set for a multitude of different flows. Furthermore, comparing figures 3.8 and 3.9 shows that these predictions are consistently more accurate than the Rolie-Poly model. There are minor disagreements in the final stages of relaxation for solutions where

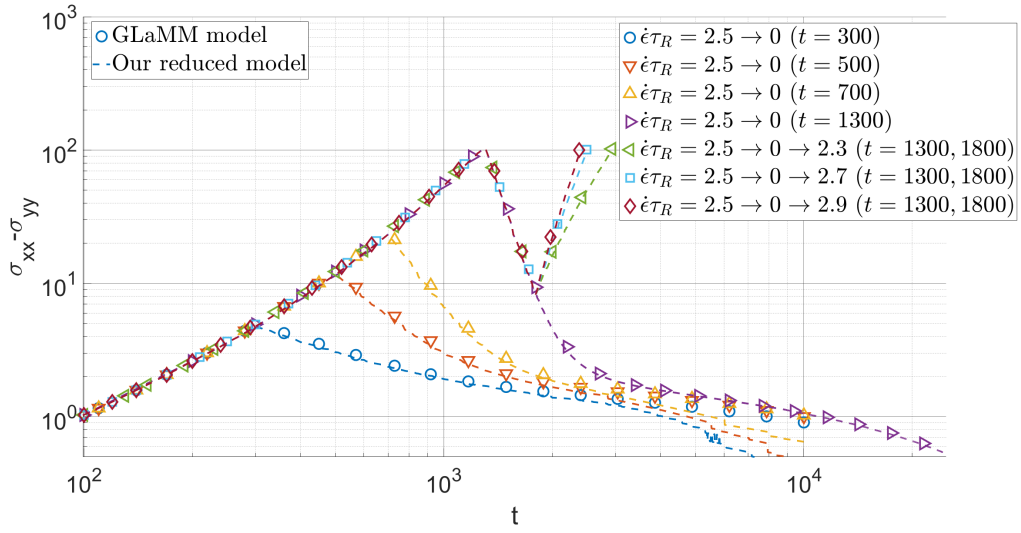


Figure 3.8: A comparison between normal stress predictions from our reduced model and the GLaMM model for uniaxial extension flows with instantaneous changes between different constant rates.

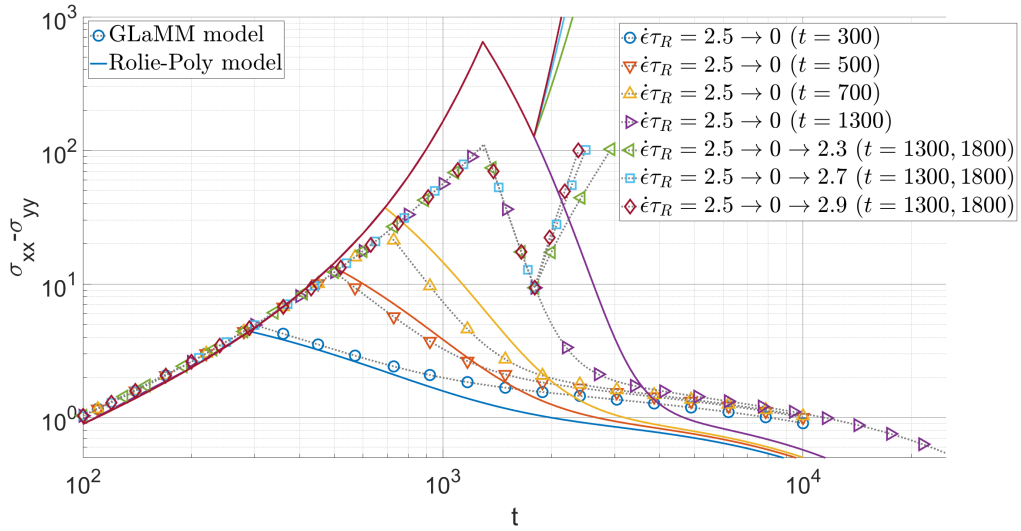


Figure 3.9: A companion plot to figure 3.8 that displays Rolie-Poly predictions on the same scale. Note that the markers for the GLaMM model have been joined with a grey dotted line for readability, this is not a separate model.

extensional flow is only present for a relatively short period of time. For a solution where the flow is present for a longer period of time such as $t = 1300$, both Γ_1 and Γ_2 grow whilst the flow is present, and they then decrease at different rates when the flow is removed. This disparity between the two functionals enables the interpolator to correctly predict relaxation. In solutions where the duration of the flow is insufficient for substantial

evolution of $\mathbf{\Gamma}$ values, both Γ_1 and Γ_2 are small during relaxation with no significant disparity between them, and the interpolator struggles to predict correct $\mathbf{f}(s, s')$ configurations. As the disagreement occurs only when both Γ_1 and Γ_2 are small, this suggests that this issue can easily be fixed with a linearised model.

3.5 Discussion of coarse-grained variable selection

This section details the methodology for the selection of functionals Γ_i. It details two different approaches. The first is an approach that uses physical intuition about polymer dynamics that is specific to the GLaMM model example. This can be applied to other models, but would require a similar level of intuition and results analysis in that model's field to replicate. Secondly, a general computational method that can automatically evaluate and select the best functional predictors, which is more readily applicable to a wide range of models. We also discuss some further comments about our current choice at the end of this section.

3.5.1 Motivation for our choice

A key difference with our approach compared to other data-driven approaches is that we utilise the underlying GLaMM model, which brings with it physical interpretations that are not possible from a purely data-driven approach. Given that we have infinitely many choices for each Γ_i we can use this insight to inform our choices. One such idea is the use of physically motivated functional choices. For example, when analysing a

polymer's dynamics some properties, such as the first normal stress difference, will be more useful than others and most of these quantities can be derived from the molecular scale dynamics contained within $\mathbf{f}(s, s', t)$. We propose that the more useful a specific functional of $\mathbf{f}(s, s')$ is in describing dynamics, the more representative it is of the overall molecular dynamics and it follows that these choices will likely be more informative when categorising $\mathbf{f}(s, s')$ configurations. Thus, if we want to predict using our reduced methodology with as few functionals as possible we should prioritise these. It is for this reason that we chose the commonly used quantity of the first normal stress difference as one of our options when we first began predicting with one functional. After testing several different choices, we found that the first normal stress difference was the best predictor from our selection. Whilst this does not rigorously prove that this is the best choice out of any possible single functional, our results suggested that prediction with two functionals would significantly improve prediction at a low computational cost. Thus we moved to utilising the reduced model with two functionals.

When predicting with two functionals, the proposed association between physical usefulness and choice suitability still applies, but it is also worth noting how the pair of functionals complement each other. Two functionals that provide similar information about the configuration of $\mathbf{f}(s, s')$ with not provide a significant improvement over prediction with one functional. For example, if we are to use the first normal stress difference as one functional for the pair with Γ_1 from equation (3.4), we can see that this formula only considers elements with $s' = s$, along what we refer to as the leading diagonal of $\mathbf{f}(s, s')$. Figure 3.2 supports this, as the shape of the arc along the plane $s = s'$ is predicted with a relatively small error, but Γ_1 fails to capture any information about the width of this f_{xx} profile along the

counter diagonal $s' = Z - s$. A functional to complement this Γ_1 should consider this fact, and this led to the creation of Γ_2 in equation (3.5).

The physical implications of the size of the counter diagonal $s' = Z - s$ are initially less clear than the relation between the leading diagonal $s = s'$ and stress response, but by analysing GLaMM model solutions, we can deduce that this trait indicates the amount of time the polymer has had to relax. A thin peak where $f_{xx}(s, Z - s)$ is small for most s suggests that the polymer melt is in the initial stages of reacting to a recent change in flow conditions, whilst a wider peak demonstrates that the various relaxation mechanisms are influencing the shape of the polymer chain, resulting in a straighter overall chain configuration with a higher correlation between the gradient vectors of distant chain elements. $\mathbf{f}(s, s')$ measures this correlation, so as the chain straightens, the values of $\mathbf{f}(s, s')$ outside of this leading diagonal $s = s'$ also become significant, when such values where negligible before the relaxation methods had taken effect. Relaxation increases the size of elements away from the main diagonal $s' = s$, which increases the value of integrals across the $s' = Z - s$ diagonal, such as the integral in equation (3.5) that we chose as Γ_2. In summary under an affine deformation, the correlation $f_{xx} - f_{yy}$ remains at zero away from the main diagonal $s = s'$. However, relaxation under flow causes this correlation to propagate outwards from the main diagonal. Thus this choice of Γ_2 is a measure of deviation from affine deformation. During a start-up flow this Γ_2 starts small and grows as the flow progresses, but at a different rate to the first normal stress difference, correcting the inability of the one-functional model to distinguish between two flows of the same $\sigma_{xx} - \sigma_{yy}$ value that are at different stages of relaxation.

To be an ideal predictor, these two functionals should be able to uniquely identify the different states of $\mathbf{f}(s, s')$, but in practice, the value of two

functionals is unlikely to be able to capture every realistic configuration of $\mathbf{f}(s, s')$. Thus a good choice should assign similar $\mathbf{f}(s, s')$ configurations to similar $\mathbf{\Gamma}$ values, and conversely, ensure that distinct $\mathbf{f}(s, s')$ profiles are assigned different values of $\mathbf{\Gamma}$. Multiple ways to confirm a "good" choice exist, from visual graphical analysis that the current choice is predicting sensible evolution behaviour when compared to the GLaMM model, to numerical calculation of an error metric. We produced graphs for and visually assessed a variety of different options to find a suitable a functional pair including stretch ratio, λ, individual elements of the stress tensor, $\gamma_{\alpha,\beta}$, Fourier modes of $\mathbf{f}(s, s')$ and even integrals over the entire $\mathbf{f}(s, s')$ surface. Graphical observations from testing these pairings established the pairing of Γ_1 and Γ_2 from equations (3.4) and (3.5) as performing the best out of our selection. In section 3.5.2 we suggest an error metric to directly quantify this discrepancy.

3.5.2 Quantifying model error

We now propose how to evaluate the suitability of a given functional set $\mathbf{f}$ for a general model. As established previously, we wish to retain an advantage in computational speed over the original PDE model, so we restrict ourselves to sets of Γ_i variables where each of these variables evolve slowly in time to allow for large time steps to be used during the evaluation of our new model. We can confirm this behaviour by assessing if $\frac{\mathrm{d}\Gamma_i}{\mathrm{d}t}$ is suitably smooth and does not change rapidly when we predict a test curve using our new model with a large time step. Now that we have only slow-moving Γ_i variables, we require a way to determine which is a better choice given two sets of h_i. A good choice for Γ_i must capture the behaviour of $\mathbf{f}$ well, minimising the error in the assumption. In order to quantify the

ability of a Γ_i to meet this criterion, we introduce metrics to measure the error in predictions of $\mathbf{f}$. For the GLaMM model example, one metric to score the different variable choices could be the root mean square error of $\mathbf{f}$,

$$\theta = \sqrt{\sum_{n=1}^{N_{\delta t}} \sum_{\alpha} \sum_{\beta} \int_{s=0}^{s=Z} \int_{s'=0}^{s'=Z} \frac{(f_{\alpha\beta}^{\mathrm{Pred}}(s, s', t_n) - f_{\alpha\beta}(s, s', t_n))^2}{4Z^2} ds ds'}, \tag{3.9}$$

where t_n is the time after n timesteps, and $N_{\delta t}$ is the total number of timesteps used for this reduced model prediction. This method to determine relative performance to the choice of functionals h_i based on metrics allows us to automate the process of choosing the best subset of functionals h_i from a wider selection by comparing the predictions from our model with results from the GLaMM model. In the GLaMM model case, we have computed the metric score for a limited collection of functional pairs, including stretch ratio λ, and verified that the first normal stress difference and its counterpart, the deviation from affine deformation, from equations (3.4) and (3.5) were the most suitable selection in this case. Since linear combinations of existing functionals are still valid choices for functionals themselves, we also explored the possibility of creating linear combinations of the highest scoring functionals from our initial small selection to further improve the fit. However, optimising the coefficients of these linear combinations to maximise the metric score reaffirmed that the best option in this situation remained the normal stress difference functional pair that we defined in equations (3.4) and (3.5). It would be possible to perform a more extensive search to potentially further minimise a chosen error metric, however our current results show very good agreement with the GLaMM model under uniaxial extension, and as such the potential improvements to model error are not necessary for the GLaMM model prediction at this

stage. Thus we postpone further development on an optimisation scheme that could perform a more extensive search for a general model and designate it as future analysis in chapter 5.2.

3.5.3 Comments on the current choice

Our current choice of functionals is unable to accurately capture very low flow rates, as can be seen in figure 3.10.

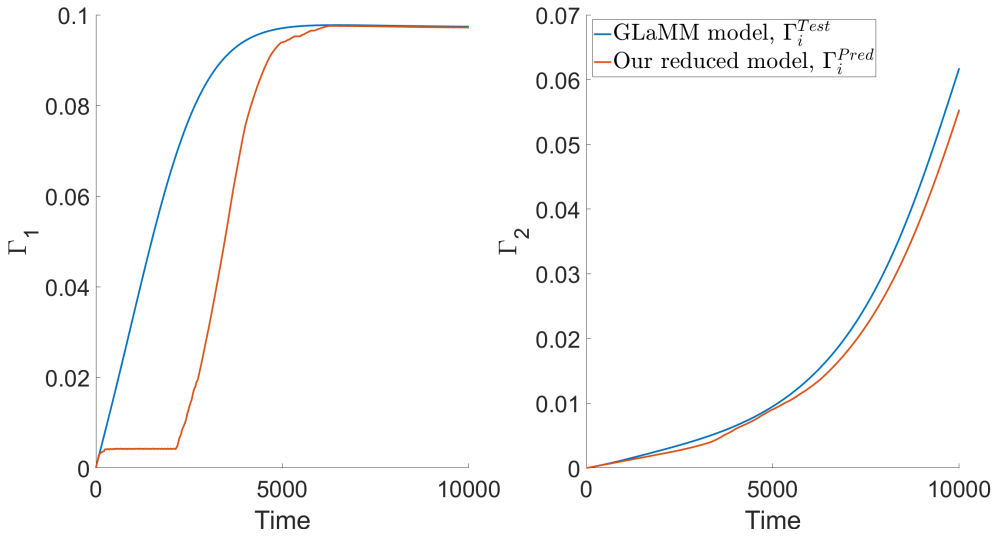


Figure 3.10: Low rate Γ_i prediction with $\dot{\epsilon}\tau_R = 0.25$, $C_\nu = 0$ and Γ_i defined in equations (3.4) and (3.5).

After some investigation we discovered that this error in low rates arises when both of our variables Γ_1 and Γ_2 are very small. Higher rates that spend very little time in this region, and lower rates with a delayed start time do not encounter this problem. We then determined that when close to the isotropic rest state, the prediction of $\frac{\mathrm{d}\Gamma_i}{\mathrm{d}t}$ is very sensitive to the correct prediction of the $\mathbf{f}$ profile. This magnifies small initial errors where the training set is less populated and leads to irrecoverable errors in the prediction of the evolution of the test case. However, this is not a major concern, as this low variable region is captured well by the Rolie-Poly

model, alleviating the need for extremely precise $\mathbf{f}$ profiles during this period. We could use the Rolie-Poly model until we leave this region, at which point our reduced model would take over. The details of improvements in low rate extension are deferred to future work in chapter 5.2. Since this problem is easily rectifiable we continue to use the same choice for our two functionals.

3.6 Overview of the model in uniaxial extension

In this chapter, we have shown how to derive and implement a reduced model that is able to accurately predict the evolution of a variety of constant and variable rate uniaxial extension flows. Specifically, the major successes of our model are as follows. Firstly, our reduced model predictions are substantially more accurate than the Rolie-Poly model, which is the leading model reduction of the GLaMM model in use today. Secondly, our reduced model is orders of magnitude faster than the GLaMM model. Thirdly, our reduced model remains accurate when the qualitative features it is asked to predict are absent in its training set. This ability to predict features the model has not explicitly been trained on is unusual for a data-driven model reduction approach and is because our methodology incorporates the underlying PDE into its methodology. Thus, it is not restricted to making estimations solely based on the data in the training set but can use the mechanics of the underlying model to make predictions about qualitatively different features. This property sets our model apart from other data-driven approaches and these advantages will persist even when the general model from section 3.1 is used with a different PDE.

We have a few remaining remarks about the small additions that were made to the general model in order to better suit the application to predicting polymer dynamics under uniaxial extension flow. Firstly, in section 3.4.3 we detailed our reasoning for the inclusion of a single transient solution with non-constant extension rate in our two-functional training set to improve the prediction accuracy of the late-stage relaxation to equilibrium. Initially this may appear to undermine our claim that our model is capable of predicting qualitatively different behaviour, but this non-constant rate is only used by our interpolator for prediction of these late-stage relaxations. The exclusion of this rate does not decrease the prediction accuracy during the change in rate and a training set of constant rates is sufficient for variable rate flows where the accuracy of late-stage relaxations is not a concern. Secondly, as mentioned in section 3.5.3 our current choice of Γ_i in equations (3.4) and (3.5) struggle to distinguish between certain $\mathbf{f}(s, s')$ configurations where both Γ_1 and Γ_2 are very small. This can occur at early times with very low uniaxial extension rates. We can easily rectify this by using the Rolie-Poly model during this initial small window and transitioning to the reduced model afterwards. Further information on the model change at low rates can be found in chapter 5.2. With these successes in predicting uniaxial extension flows, we now further develop our model for the prediction of shear flows in chapter 4.

Chapter 4

Reduced model with shear flows

In the previous research chapter, chapter 3, we defined our new reduced model for the GLaMM model and applied it to uniaxial extension flows. This reduced model uses slow-moving degrees of freedom to predict the full solution and reduce the GLaMM model PDE. These degrees of freedom are defined in terms of functionals, h_i, of the PDE solution, $\mathbf{f}(s, s', t)$, so that $\Gamma_i = h_i(\mathbf{f})$. With suitable functional choices, we recover a prediction for the full solution, $\mathbf{f}^{\text{Pred}}$, from knowledge of Γ_i values by way of interpolation from a set of training data. By using $\Gamma_1 = \frac{12G_e}{5Z^2} \int_0^Z f_{xx}(s, s) - f_{yy}(s, s)\ ds$ and $\Gamma_2 = \frac{12G_e}{5Z^2} \int_0^Z f_{xx}(s, Z-s) - f_{yy}(s, Z-s)\ ds$, with a four point quadrant interpolator, we produced accurate predictions for uniaxial extension flows of various rates.

In this chapter, we adapt our new model to predict polymer shear flows. We discuss the extra considerations when a polymer model includes rotational elements in section 4.1. With these alterations to the model, we predict the evolution of shear flows in section 4.2, using the same functionals, in-

terpolator and training data as were used to predict uniaxial extension flows. With evidence to suggest that two functionals would be insufficient to capture all the features of shear flows, we introduce a basic choice for a third functional, h_3, in section 4.3 to capture this extra degree of freedom. The introduction of another functional affects the training data and the method of interpolation, this is explored in section 4.4. With the introduction of a revised training set and interpolator for three functionals, we now predict shear flows in section 4.5. These shear flow predictions using three functionals are an improvement to prediction using two, despite the initial choice for the third functional causing the prediction to incur significant numerical errors. Section 4.6 explores the source of this noise and establishes a revised third functional with this new information. During the investigation into the noise, we saw the potential for an improvement to our interpolator which could further reduce the noise, the details and effects of which are presented in section 4.7. With our model now predicting one specific shear flow to a suitable degree of accuracy, we evaluate the performance of our model for various other constant and non-constant shear flows in section 4.8. The final section, section 4.11, provides an overview of the major results in this chapter.

4.1 Adapting the model to flows with rotation

4.1.1 Principal direction

The primary difference that shear flows have compared to uniaxial extension flows is the concept of the rotation of the chain. In this section we

define the principal direction of a polymer melt and explain why such an introduction is necessary. This principal direction is the quantity we use to track the average rotation of polymer chains within the melt. When approximating solutions of the GLaMM model PDE, the unknown is the tensor of functions $\mathbf{f}(s, s', t)$. We formulate our definition for the principal direction in terms of components of the stress tensor, where the stress tensor is calculated as $\boldsymbol{\sigma}(t) = \frac{12G_e}{5Z} \int_0^Z \mathbf{f}(s, s, t)ds$. The principal direction is defined as the direction of the largest principal stress. These principal stresses are the values of the normal stresses, σ_{11}, σ_{22} and σ_{33}, when the coordinates are rotated so that shear stresses, σ_{ij} for $i \neq j$ are zero. In practice, principal stress is found by diagonalising the stress tensor by using its eigenvectors. The resulting eigenvalues are then the principal stresses, and the eigenvector corresponding to the largest eigenvalue is the principal direction.

$$\begin{pmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{xy} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{xz} & \sigma_{yz} & \sigma_{zz} \end{pmatrix} \rightarrow \begin{pmatrix} \sigma_{11} & 0 & 0 \\ 0 & \sigma_{22} & 0 \\ 0 & 0 & \sigma_{33} \end{pmatrix}$$

In equilibrium, the polymer melt is in its isotropic rest state with $\boldsymbol{\sigma} \propto \mathbf{I}$. In this state, all eigenvalues are equal, and any direction is an eigenvector. Thus all vectors can be considered as the principal direction for the isotropic rest state. A uniaxial extension flow will have a constant principal direction, corresponding to the direction of the flow. Shear flows do not have a constant principal direction, the chains will have a principal direction that rotates. Shear flows in the xy-plane will always have principal directions of the form $(a, b, 0)$ but the values of a and b will change over time.

Knowledge of this principal direction will enable data from a flow in one principal direction to predict similar flows in any other principal direction without additional training data. For example, let us compare a uniaxial

extension in the x-direction with one in the y-direction. For an isotropic region of the material, these two extensional flows will elicit fundamentally the same response, and by rotating one set of results by 90° they become identical. Our current model would not be able to predict the result of a uniaxial extension flow in the y-direction with the training set we have used until now, as the principal directions of the test flow to be predicted differs from that of the training data from uniaxial extension flows in the x-direction. However, if we know the current principal direction of the test flow, and the training data has a uniform, known principal direction, then we can predict this flow if we add steps to the model to perform the relevant rotation transformations. Shear flow has a non-constant principal direction, but provided we know the principal direction at the current time, training data with a uniform principal direction could predict this as well.

In order to track this principal direction, we introduce a new tensorial functional $\mathbf{S}(t)$ to denote the normalised end-to-end vector, sometimes denoted as $\langle \mathbf{uu} \rangle$ in other literature, such that,

$$\mathbf{S}(t) = \langle \mathbf{uu} \rangle = \frac{\int_0^Z \mathbf{f}(s,s,t)ds}{\mathrm{Tr}\left(\int_0^Z \mathbf{f}(s,s,t)ds\right)} = \frac{\boldsymbol{\sigma}(t)}{\mathrm{Tr}\left(\boldsymbol{\sigma}(t)\right)}, \tag{4.1}$$

$$\begin{aligned} \frac{\mathrm{d}\mathbf{S}(t)}{\mathrm{d}t} &= \frac{1}{\mathrm{Tr}\left(\int_0^Z \mathbf{f}(s,s,t)ds\right)} \int_0^Z \frac{\partial \mathbf{f}(s,s,t)}{\partial t} ds \\ &\quad - \frac{\mathrm{Tr}\left(\int_0^Z \frac{\partial \mathbf{f}(s,s,t)}{\partial t} ds\right)}{\left(\mathrm{Tr}\left(\int_0^Z \mathbf{f}(s,s,t)ds\right)\right)^2} \int_0^Z \mathbf{f}(s,s,t)ds. \end{aligned} \tag{4.2}$$

Calculating the principal direction using $\mathbf{S}$ will return the same result as $\boldsymbol{\sigma}$, since $\mathbf{S} \propto \boldsymbol{\sigma}$, but using $\mathbf{S}$ simplifies some later calculations since $\mathrm{Tr}\left(\mathbf{S}\right) = 1$. With equation (4.1), we are now in a position to modify our previous algorithm so that it considers the rotation of the flow when predicting

$\mathbf{f}(s, s', t)$.

4.1.2 Defining the laboratory and principal frames of reference

In this section, we detail changes to the algorithm in order to predict flows with various rotations using a training set with one uniform principal direction. Firstly, for us to be able to predict an $\mathbf{f}(s, s', t)$ for a general rotation, our training set must have a known, uniform principal direction. We create our training set so that all training data has principal direction $(1, 0, 0)$.

For the sake of clarity, let us define two different frames of reference for this new methodology. The laboratory frame shall refer to test data or predicted data that incorporates rotation. The principal frame on the other hand shall refer to any data that has had its rotation "removed" by a transformation so that its principal direction is $(1, 0, 0)$. All of our training data will be in the principal frame. To return to the example in section 4.1.1, two uniaxial extension flows in different directions will be different when viewed in the laboratory frame, but equivalent if both are transitioned into the principal frame. The normalised end-to-end vector $\mathbf{S}$ from equation (4.1) will later be used to define the transformation between this laboratory frame and the principal frame. We denote these different frames of reference with subscripts, $\mathbf{f}_{\text{Prin}}$ and $\mathbf{f}_{\text{Lab}}$. Most variables such as Γ_i and $\mathbf{S}$ for this methodology are only defined in one of the two reference frames with only $\mathbf{f}$ defined in both frames. Our functionals will always be defined in the principal frame, $\Gamma_i = h_i(\mathbf{f}_{\text{Prin}})$, because to predict solutions with any principal direction from a singular training set, the result from our functionals must be rotationally invariant. In contrast, $\mathbf{S}$ will always be defined in the laboratory frame, as we will use $\mathbf{S}$ to track rotation.

The formulae for the functionals h_i require a subtle change to generalise them to be applicable to variables in the principal frame. Here we alter equations 3.4 and 3.5 in order to apply them to the principal frame.

$$\Gamma_1 = \frac{12G_e}{5} \int_0^Z (\mathbf{f}_{\text{Prin}}(s,s))_{11} - (\mathbf{f}_{\text{Prin}}(s,s))_{22}\ ds, \tag{4.3}$$

$$\Gamma_2 = \frac{12G_e}{5} \int_0^Z (\mathbf{f}_{\text{Prin}}(s,Z-s))_{11} - (\mathbf{f}_{\text{Prin}}(s,Z-s))_{22}\ ds. \tag{4.4}$$

As these equations 4.3 and 4.4 act upon $\mathbf{f}_{\text{Prin}}$ in the principal frame, they are rotationally invariant. Uniaxial extension flows in any direction of the same rate will show the same evolution for Γ_1 and Γ_2. However, for general flows, Γ_1 will no longer represent the first normal stress difference. It is still possible to recover the value of the first normal stress difference from equation (3.4) using $\mathbf{f}_{\text{Lab}}$. With the functionals and training set now in the principal frame but the GLaMM model PDE only defined in the laboratory frame, a transformation between these two frames of reference must be defined.

To transition between $\mathbf{f}_{\text{Lab}}$ and $\mathbf{f}_{\text{Prin}}$ we introduce the transformation tensor $\mathbf{V}$ in the laboratory frame. If $\mathbf{D}$ is defined as a diagonal matrix with the eigenvalues of $\mathbf{S}$ in descending magnitude along its diagonal, then $\mathbf{V}$ is a tensor of eigenvectors such that $\mathbf{S}\cdot\mathbf{V} = \mathbf{V}\cdot\mathbf{D}$. An explicit formula for this tensor of ordered eigenvectors, $\mathbf{V}$, for the 2D case can be seen in equation (C.4). To transition $\mathbf{f}$ between the two frames of reference, we transform $\mathbf{f}$ by rotation using $\mathbf{f}_{\text{Lab}} = \mathbf{V}\cdot\mathbf{f}_{\text{Prin}}\cdot\mathbf{V}^T$. When adding data to the training set we can use the inverse transformation $\mathbf{f}_{\text{Prin}} = \mathbf{V}^T\cdot\mathbf{f}_{\text{Lab}}\cdot\mathbf{V}$ to ensure that all training data has principal direction $(1,0,0)$.

4.1.3 Changes to the method

Now let us assume that we know the values of the variables[1] from the functionals Γ_i^{Pred}, and the tensor $\mathbf{S}^{\text{Pred}}$ at time $t = T$. Knowledge of $\mathbf{S}^{\text{Pred}}$ means we can also calculate $\mathbf{V}^{\text{Pred}}$ at the current time. We now define an algorithm that uses this knowledge to determine Γ_i^{Pred} and $\mathbf{S}^{\text{Pred}}$ at time $t = T + \delta t$. When under shear, the direction of the principal frame depends on time and thus derivatives of the transformation must be accounted for when transforming $\frac{\partial \mathbf{f}}{\partial t}$ back into the principal frame. Full details and formulae for this are in appendix C.

1. Calculate $\mathbf{f}_{\text{Prin}}^{\text{Pred}}$ from Γ_i^{Pred}.

   (a) Interpolate $\mathbf{f}_{\text{Prin}}^{\text{Pred}}$ from $\mathbf{f}_{\text{Prin}}^{\text{Train}}$ at Γ_i^{Pred} at $t = T$.

   [Principal frame]

2. Transform $\mathbf{f}_{\text{Prin}}^{\text{Pred}}$ to $\mathbf{f}_{\text{Lab}}^{\text{Pred}}$ using eigenvectors.

   (a) Find the matrix of eigenvectors, $\mathbf{V}^{\text{Pred}}$, of the current rotation, $\mathbf{S}^{\text{Pred}}$, at $t = T$.

   [Laboratory frame]

   (b) Calculate $\mathbf{f}_{\text{Lab}}^{\text{Pred}}$ by rotating this $\mathbf{f}_{\text{Prin}}^{\text{Pred}}$ into the laboratory frame using $\mathbf{V}^{\text{Pred}}$ at $t = T$.

   [Principal frame $\rightarrow$ Laboratory frame]

3. Calculate $\frac{\partial}{\partial t}\mathbf{f}_{\text{Lab}}^{\text{Pred}}$ using the GLaMM model.

   (a) Use the GLaMM model (2.3) to calculate $\frac{\partial}{\partial t}\mathbf{f}_{\text{Lab}}^{\text{Pred}}$ from $\mathbf{f}_{\text{Lab}}^{\text{Pred}}$ at $t = T$.

   [Laboratory frame]

[1] Superscript "Pred", "Test" and "Train" attached to variables refer to if they are derived from new model predictions, test data from the GLaMM model or training set data respectively.

4. Transform $\frac{\partial}{\partial t}\mathbf{f}^{\text{Pred}}_{\text{Lab}}$ back to $\frac{\partial}{\partial t}\mathbf{f}^{\text{Pred}}_{\text{Prin}}$ using eigenvectors and the required derivatives.

   (a) Calculate $\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{S}^{\text{Pred}}$ from $\mathbf{f}^{\text{Pred}}_{\text{Lab}}$ and $\frac{\partial}{\partial t}\mathbf{f}^{\text{Pred}}_{\text{Lab}}$ at $t = T$, using equation (4.2).

   [Laboratory frame]

   (b) Calculate $\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{V}^{\text{Pred}}$ from $\mathbf{S}^{\text{Pred}}$ and $\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{S}^{\text{Pred}}$ at $t = T$, using equation (C.5).

   [Laboratory frame]

   (c) Calculate $\frac{\partial}{\partial t}\mathbf{f}^{\text{Pred}}_{\text{Prin}}$ from $\mathbf{f}^{\text{Pred}}_{\text{Lab}}$, $\frac{\partial}{\partial t}\mathbf{f}^{\text{Pred}}_{\text{Lab}}$, $\mathbf{V}^{\text{Pred}}$ and $\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{V}^{\text{Pred}}$ at $t = T$, using equation (C.3).

   [Laboratory frame → Principal frame]

5. Calculate $\frac{\mathrm{d}\Gamma_i}{\mathrm{d}t}^{\text{Pred}}$ from $\mathbf{f}^{\text{Pred}}_{\text{Prin}}$ and $\frac{\partial}{\partial t}\mathbf{f}^{\text{Pred}}_{\text{Prin}}$.

   (a) Calculate each $\frac{\mathrm{d}}{\mathrm{d}t}\Gamma_i^{\text{Pred}}$ from $\mathbf{f}^{\text{Pred}}_{\text{Prin}}$ and $\frac{\partial}{\partial t}\mathbf{f}^{\text{Pred}}_{\text{Prin}}$ at $t = T$, using equations (C.1) and (C.2).

   [Principal frame]

6. Take a timestep in both Γ_i^{Pred} and $\mathbf{S}^{\text{Pred}}$.

   (a) Utilise the forward Euler method to find the values of each Γ_i^{Pred} and $\mathbf{S}^{\text{Pred}}$ at $t = T + \delta t$.

   [Principal and laboratory frames]

At first glance, this method may seem more complex than the formulation without rotation in section 3.1, however, there are only two major changes. The first major variation in the method with rotation is that, unlike before, the GLaMM model calculation and the training set interpolation must be performed in different frames of reference. This necessitates two additional intermediate steps 2 and 4 to calculate and apply the transformations required to transition between these frames of reference. The other necessary

adjustment is related to this; we now track the evolution of a rotation variable, $\mathbf{S}$, alongside our other variables Γ_1 and Γ_2. The value of $\mathbf{S}$ is required to transform between $\mathbf{f}_{\text{Lab}}$ and $\mathbf{f}_{\text{Prin}}$. We require the ability to switch between frames of reference freely at each timestep, and so we will require knowledge of its evolution throughout time alongside each Γ_i. Fortunately, as $\mathbf{S}$ is also a functional of $\mathbf{f}$ and so it can be treated similarly to the other functionals during evaluation and evolution, the largest distinction being that $\mathbf{S} = h_{\mathbf{S}}(\mathbf{f}_{\text{Lab}})$ must be calculated in the laboratory frame and $\Gamma_i = h_i(\mathbf{f}_{\text{Prin}})$ must instead be handled in the principal frame.

We now provide an overview detailing each step. Firstly, this formulation mandates that certain steps must be performed in the principal frame and laboratory frames respectively, but the principal frame and laboratory frames are equivalent for uniaxial extension flow in the x-direction and so this generalised method simplifies to the formulation in section 3.1.1 in this case. Step 1 is the same 2D interpolation as before, only with the extra requirement that any further data we wish to add to the training set must be rotated into the principal frame first. Step 2 calculates the correct transformation matrix using eigenvectors, and then transforms the interpolation predictions from the principal frame to the laboratory frame. With a prediction of $\mathbf{f}$ in the laboratory frame, we now apply the GLaMM model from equation (2.3) in step 3. Since all functionals are now defined only in the principal frame, we must transform this laboratory frame prediction of $\frac{\partial \mathbf{f}}{\partial t}$ back to the principal frame before we can calculate $\frac{\mathrm{d}\Gamma_i}{\mathrm{d}t}$. Step 4 calculates the required transformation derivative, $\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}t}$, and then performs this transformation back to the principal frame. Step 5 evaluates the derivatives of the functionals using equations (C.1) and (C.2). Step 6 predicts the values at the next timestep using the forward differencing Euler method on each functional, Γ_i, but the methodology with rotation also predicts $\mathbf{S}$ at

the next timestep as we need to know the correct transformation between frames of reference at every timestep. Euler timestepping was chosen for its computation speed, but any forward timestepping method could be used. There was no need to calculate the derivative $\frac{\mathrm{d}\mathbf{S}}{\mathrm{d}t}$ during step 5, as it was previously calculated during the transformation back to the principal frame in step 4a. By performing repeated iterations of this process, we are now able to predict the evolution of flows in any principal direction, rather than being restricted to the principal direction which matches our training data.

4.2 Shear prediction with two functionals

Now that we have established a method that can predict flows or varying rotation in section 4.1.3, we use the training set from chapter 3 to predict shear flows. This training set consists of uniaxial extension flows in the x-direction with extension rates $\dot{\epsilon}\tau_R = 0.1$ to $\dot{\epsilon}\tau_R = 3$, with one additional relaxation flow where the flow is removed, $\dot{\epsilon}\tau_R = 0$, partway through a uniaxial extension of rate $\dot{\epsilon}\tau_R = 2.1$. This relaxation flow was introduced in section 3.4. Uniaxial extensions in the x-direction have a constant principal direction of $(1, 0, 0)$, so this training set is already in the principal frame and no alterations need to be made.

During the prediction of uniaxial flows in chapter 3 we ignore the effects of convective constraint release (CCR) on the GLaMM model. This is acceptable as CCR is negligible in uniaxial extension flows, but this is not the case for shear flows. We begin prediction without CCR as our current training set of uniaxial extension data may produce more accurate predictions without CCR. We will then move on to more realistic shear flows with $C_\nu \neq 0$. We further discussed the details of the relaxation

methods including CCR during the GLaMM model introduction in section 2.3. In general whenever we calculate shear flows with the GLaMM model from equation (2.3), we denote the shear rate as $\dot{\gamma}$, and set $\kappa_{xy} = \dot{\gamma}, \kappa_{\alpha\beta} = 0 \; \forall \; \alpha\beta \neq xy$ with $C_\nu = 0.1$ unless otherwise specified.

We will first use the two functionals defined in equations (4.3) and (4.4) to predict a shear flow with rate $\dot{\gamma}\tau_R = 2$ and without CCR, $C_\nu = 0$. Results can be seen in figures 4.1 and 4.2. Different colour lines refer to the different frames of reference in which the calculation is performed. Blue and orange lines are from calculations in the principal frame, whilst green and red are calculations in the laboratory frame. Note that we delay the introduction of our reduced model for a single timestep, using delayed start time $T = 10$, as described in section 3.2. This means that we calculate with the full GLaMM model until $t = T$, using the initial condition of $\mathbf{f}^{\text{Pred}}(s, s', 10) = \mathbf{f}^{\text{Test}}(s, s', 10)$ for our reduced model. This is to avoid the calculation of $\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}t}$ at $\mathbf{f}_{eq}$ as this is undefined, as further discussed in appendix C. S_{yy} is not shown in figure 4.2 as this can be calculated from S_{xx}.

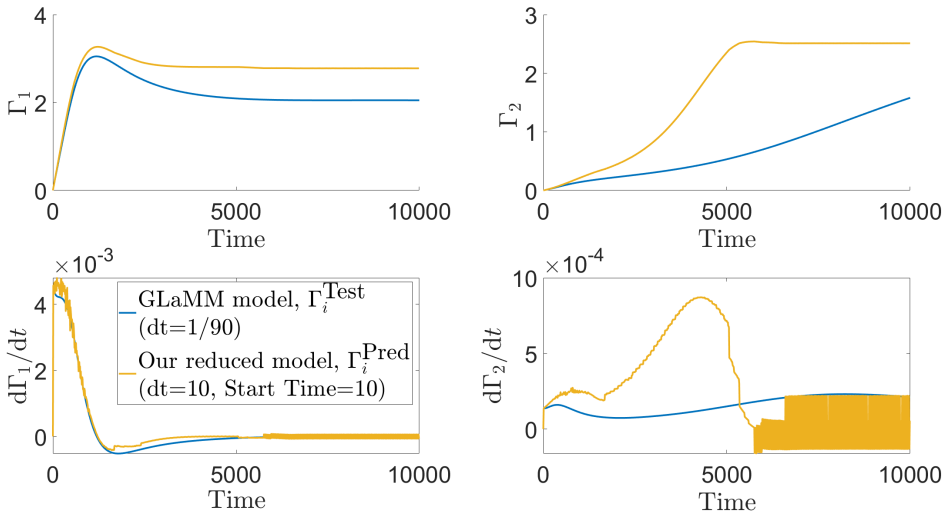


Figure 4.1: Reduced model predictions of Γ_i and $\frac{\mathrm{d}\Gamma_i}{\mathrm{d}t}$ in the principal reference frame at $\dot{\gamma}\tau_R = 2$ and $C_\nu = 0$.

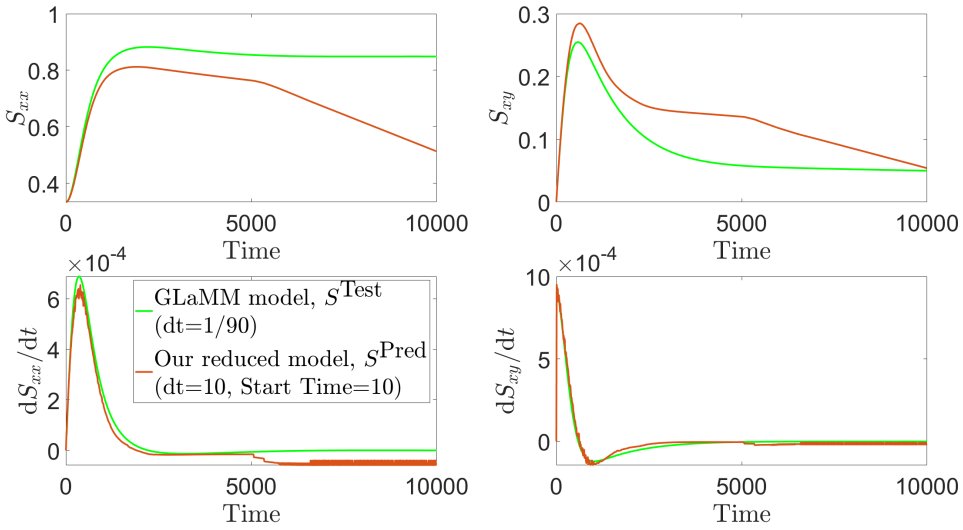


Figure 4.2: Reduced model predictions of $\mathbf{S}$ and $\frac{\mathrm{d}\mathbf{S}}{\mathrm{d}t}$ in the laboratory reference frame at $\dot{\gamma}\tau_R = 2$ and $C_\nu = 0$.

As can be seen in the figures 4.1 and 4.2, even without the effects of CCR small errors in the prediction of $\frac{\mathrm{d}\Gamma_1}{\mathrm{d}t}$ and $\frac{\mathrm{d}\mathbf{S}}{\mathrm{d}t}$ cause Γ_1^{Pred} and $\mathbf{S}^{\mathrm{Pred}}$ to drift away from Γ_1^{Test} and $\mathbf{S}^{\mathrm{Test}}$ due to the accumulation of timestepping errors over time Γ_2^{Pred} and $\frac{\mathrm{d}}{\mathrm{d}t}\Gamma_2^{\mathrm{Pred}}$ incur significant errors, which would suggest that this pair of functionals is not suitable for shear flow prediction. The accuracy of Γ_1^{Pred}, Γ_2^{Pred} and $\mathbf{S}^{\mathrm{Pred}}$ depend on each other, as errors in one of these variables will be inherited by the others on the next timestep. In order to isolate which elements are causing errors, let us fix $\mathbf{S}$ to its true value, $\mathbf{S}^{\mathrm{Pred}} = \mathbf{S}^{\mathrm{Test}}$ throughout the evolution, and only predict Γ_1 and Γ_2. We choose to set $\mathbf{S}$ because a small change in the transformation matrix between frames of reference could lead to drastically different $\mathbf{f}^{\mathrm{Pred}}$. Predicting the same flow with $\mathbf{S}^{\mathrm{Pred}}$ fixed produces figure 4.3.

Despite noticeable inaccuracies in the prediction of Γ_2, figure 4.3 suggests that the accuracy of both quantities increases significantly when the true rotation, $\mathbf{S}^{\mathrm{Test}}$, is known. In particular, we now have an accurate evolution for Γ_1^{Pred}. This supports our hypothesis that the majority of errors in figure 4.1 arise from an erroneous prediction of the rotation tensor to transi-

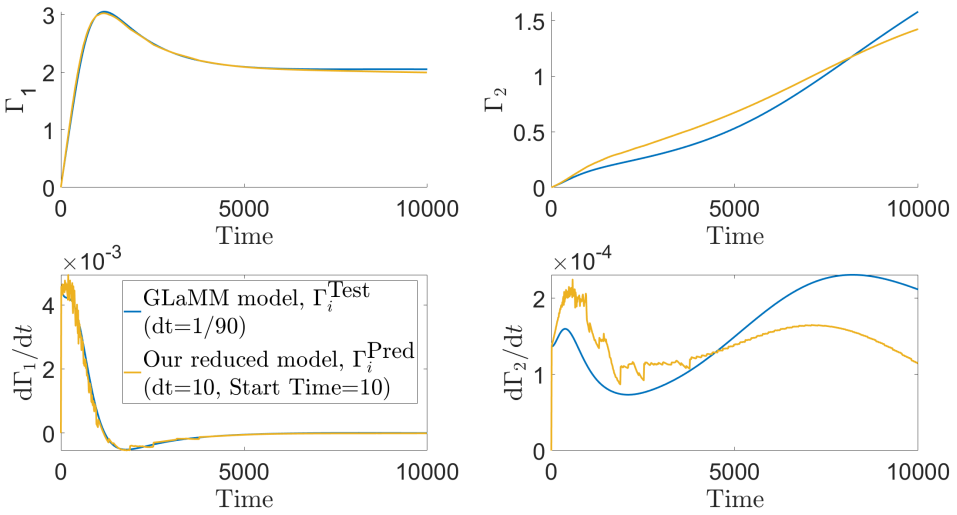


Figure 4.3: Reduced model predictions of Γ_i and $\frac{d\Gamma_i}{dt}$ in the principal reference frame at $\dot{\gamma}\tau_R = 2$ and $C_\nu = 0$. $\mathbf{S}^{\text{Pred}}$ is fixed at the true value.

tion between reference frames. Even then, Γ_2^{Pred} still has significant errors. These errors prevent the reduced model with this functional choice from being applicable to shear prediction, but also suggest the way to proceed without changing the model further.

4.3 Introduction of a third functional

4.3.1 Motivation for a third functional

Figure 4.3 shows significant difficulty in the prediction of shear flow with the current two functionals. Since any errors in principal direction are removed for figure 4.3, this confirms that the problem stems from the inadequacy of the current functionals to predict $\frac{\partial \mathbf{f}}{\partial t}$, and thus $\frac{d\mathbf{S}}{dt}$. This error in the evolution of principal direction does further compound these errors, but if a choice of functionals is not suitable to uphold our assumption, it will cause errors in the evolution of all quantities in the reduced model. Thus we use figure 4.3 to identify a better choice to reduce both errors.

The error in Γ_2^{Pred} whilst all other quantities are predicted accurately or fixed to the true values could imply one of two things. Either the current Γ_2 choice is not providing much information about the characteristics of $\mathbf{f}$ for shear flows, this is supported by the accuracy of Γ_1^{Pred} regardless of the error in Γ_2^{Pred}, or that the possible configurations of $\mathbf{f}$ are more nuanced in shear flows and any two functionals would not be sufficient to capture the new characteristics. If Γ_2 does not provide enough information about shear flows, a different choice of h_2 might be appropriate, but if we find that any choice of two functionals would be insufficient to capture all features of $\mathbf{f}$ under shear flow, a different approach is required. Another angle to verify if two functionals are insufficient to capture shear flows is to confirm the existence of two physically attainable $\mathbf{f}(s, s')$ profiles with significantly different configurations, but which take the same Γ_1 and Γ_2 values, invalidating the assumption that we can categorise all feasible $\mathbf{f}(s, s')$ profiles with that specific choice. With further exploration of possible choices for two functionals, we couldn't find a suitable Γ_2 that allows us to predict both uniaxial extensions and shear flows well with the same choice.

Whilst it may be possible for a pair of functionals to be able to satisfy our assumption, our analysis of two-functional shear flow prediction suggests that the existence of such a pair is unlikely and so we choose instead to extend the number of functionals to three as this is within the scope of our reduced model without changes to the methodology. This would allow us the freedom to categorise another feature of $\mathbf{f}(s, s')$ configurations, improving prediction accuracy whilst not significantly affecting computational complexity. Since this choice of two functionals can predict uniaxial extension flows well but not shear flows, any new Γ_3 we introduce would need to be able to predict $\mathbf{f}(s, s')$ configurations from shear flows well, and thus naturally we will need to introduce shear training data into the train-

ing set. Further details on this inclusion can be found in section 4.4.1, but first we establish what our choice for this third functional is. We want to introduce a third functional that distinguishes shear from extension data and provides some measure of the extent of shear. Let us consider the form of $\mathbf{f}(s, s')$ configurations in the principal frame for polymer dynamics in 2D. For $\mathbf{f}_{\text{Prin}}$ we have 3 distinct components, $(\mathbf{f}_{\text{Prin}})_{11}$, $(\mathbf{f}_{\text{Prin}})_{22}$ and $(\mathbf{f}_{\text{Prin}})_{12}$ due to symmetry in $(\mathbf{f}_{\text{Prin}})_{\alpha\beta} = (\mathbf{f}_{\text{Prin}})_{\beta\alpha}$. $(\mathbf{f}_{\text{Prin}}(s, s'))_{12}$ is zero for all s and s' for any extensional flows. $(\mathbf{f}_{\text{Prin}})_{12}$ is also a quantity that our choices for Γ_1 and Γ_2 never consider in their calculations for this reason and thus are unable to capture. In shear however, the average of $(\mathbf{f}_{\text{Prin}})_{12}$ along s=s' is zero by definition, but $(\mathbf{f}_{\text{Prin}})_{12}$ is not zero for all values of s and s'. For this reason, our initial choice for a third functional functional in the following section is based on the variance of $(\mathbf{f}_{\text{Prin}})_{12}$.

4.3.2 Basic choice for a third functional

We choose to use the variation about the principal direction as the third functional. This is chosen as an initial choice as this will be zero throughout any uniaxial extension flows, and variation about the principal direction is a suitable quantity to track how effective shear is at the current time. Let $\mathbf{V}_A(t)$ be a vector denoting the principal direction of a polymer chain. $\mathbf{V}_A(t)$ is the first column of $\mathbf{V}$ from equation (C.4). Then define $\mathbf{V}_{B(s)}(s, t)$ as the local principal direction. This is calculated in much the same way as the global principal direction, but at a specific $s = s_1$, so that $\mathbf{V}_{B(s_1)}(s_1, t)$ is the unit eigenvector corresponding to the largest eigenvalue of $\frac{\mathbf{f}(s_1,s_1,t)}{\text{Tr}\,(\boldsymbol{\sigma}(t))}$. The equation for $\mathbf{V}_{B(s)}$ can be written as,

$$\mathbf{V}_{B(s)}(s, t) = \left(\frac{a(s, t)}{\sqrt{a^2(s, t) + b^2(s, t)}}, \frac{b(s, t)}{\sqrt{a^2(s, t) + b^2(s, t)}} \right), \tag{4.5}$$

where,

$$a(s,t) = \begin{cases} \lambda_L(s,t) - L_{yy}(s,t) & \text{for} \quad L_{yy}(s,t) < L_{xx}(s,t) \\ L_{xy}(s,t) & \text{for} \quad L_{xx}(s,t) < L_{yy}(s,t) \end{cases},$$

$$b(s,t) = \begin{cases} L_{xy}(s,t) & \text{for} \quad L_{yy}(s,t) < L_{xx}(s,t) \\ \lambda_L(s,t) - L_{xx}(s,t) & \text{for} \quad L_{xx}(s,t) < L_{yy}(s,t) \end{cases},$$

$$\lambda_L(s,t) = \frac{L_{xx}(s,t) + L_{yy}(s,t) + \sqrt{(L_{xx}(s,t) - L_{yy}(s,t))^2 + 4L_{xy}^2(s,t)}}{2},$$

$$L_{\alpha\beta}(s,t) = \frac{f_{\alpha\beta}(s,s,t)}{\mathrm{Tr}\,(\boldsymbol{\sigma}(t))}.$$

Note that in equation (4.5) we write $\mathbf{V}_{B(s)}$ as a function dependant on s and t, but as each term is only dependant on the elements of $\mathbf{f}(s,s,t)$, knowledge of $\mathbf{f}$ is sufficient to calculate $\mathbf{V}_{B(s)}$ and this equation can be formulated as $\mathbf{V}_{B(s)}(\mathbf{f}(s,s,t))$.

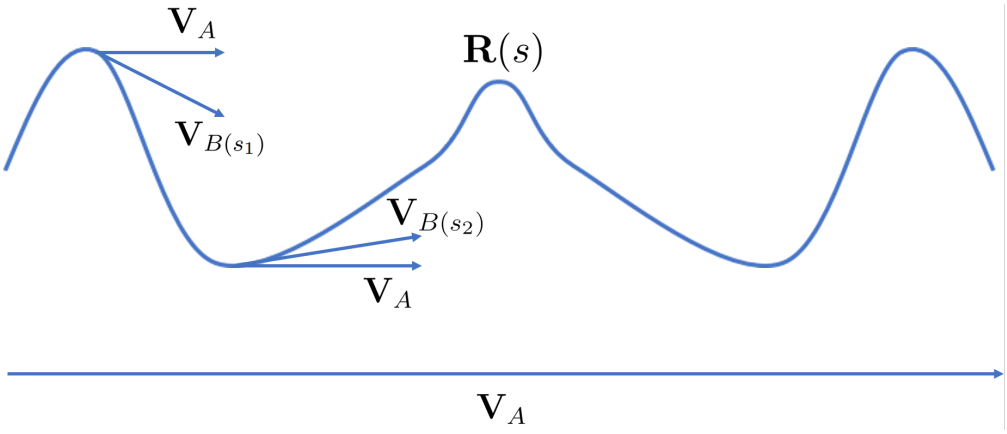


Figure 4.4: A visualisation of a polymer chain, $\mathbf{R}(s)$, showing the difference between principal direction, $\mathbf{V}_A(t)$, and local principal direction, $\mathbf{V}_{B(s)}(s,t)$.

The variation in principal direction can then be defined as variance in the angle between the unit vectors $\mathbf{V}_A(t)$ and $\mathbf{V}_{B(s)}(s,t)$ in the laboratory frame this is defined as,

$$\left[\theta_{PD}(s,t)\right]_{\text{Lab}} = \cos^{-1}\left(\left[\mathbf{V}_A(t)\right]_{\text{Lab}} \cdot \left[\mathbf{V}_{B(s)}(s,t)\right]_{\text{Lab}}\right).$$

If the local and average principal direction are the same for all s then the variance in the angle will be zero. This is true under uniaxial extension flows, but not shear flows. When in the principal frame, $\left[\mathbf{V}_A(t)\right]_{\text{Prin}} = \hat{\mathbf{x}}$ by definition, and since θ_{PD} is unaffected by a global rotation, $\left[\theta_{PD}\right]_{\text{Lab}} = \left[\theta_{PD}\right]_{\text{Prin}}$. Due to this from now on we drop the notation for frame of reference and simply write θ_{PD}. And so the variance of $\cos(\theta_{PD}(s,t))$ over s is,

$$\begin{aligned} Var\big(\cos(\theta_{PD}(s,t))\big) &= Var\Big(\hat{\mathbf{x}} \cdot \left[\mathbf{V}_{B(s)}(s,t)\right]_{\text{Prin}}\Big) \\ = \int_0^Z \Big(\hat{\mathbf{x}} \cdot \left[\mathbf{V}_{B(s)}(s,t)\right]_{\text{Prin}}\Big)^2 ds &- \Big(\int_0^Z \hat{\mathbf{x}} \cdot \left[\mathbf{V}_{B(s)}(s,t)\right]_{\text{Prin}} ds\Big)^2. \end{aligned}$$

Using this to inform our choice, we define the third functional as,

$$\begin{aligned} \Gamma_3 = h_3(\mathbf{f}_{\text{Prin}}(s,s,t)) &= \int_0^Z 1 - \Big(\hat{\mathbf{x}} \cdot \left[\mathbf{V}_{B(s)}(s,t)\right]_{\text{Prin}}\Big)^2 ds \\ &= \int_0^Z \sin^2(\theta_{PD}(s,t))\, ds, \end{aligned} \tag{4.6}$$

with respective derivative,

$$\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t} = \int_0^Z -2\Big(\hat{\mathbf{x}} \cdot \left[\mathbf{V}_{B(s)}(s,t)\right]_{\text{Prin}}\Big)\Big(\hat{\mathbf{x}} \cdot \frac{\partial}{\partial t}\left[\mathbf{V}_{B(s)}(s,t)\right]_{\text{Prin}}\Big)\, ds, \tag{4.7}$$

where,

$$\frac{\partial}{\partial t}\mathbf{V}_{B(s)}(s,t) = \frac{a'(s,t)b(s,t) - b'(s,t)a(s,t)}{(a^2(s,t) + b^2(s,t))^{\frac{3}{2}}}(b(s,t),\ -a(s,t)), \tag{4.8}$$

$$a'(s,t) = \begin{cases} \lambda'_L(s,t) - L'_{yy}(s,t) & \text{for} \quad L_{yy}(s,t) < L_{xx}(s,t) \\ L'_{xy}(s,t) & \text{for} \quad L_{xx}(s,t) < L_{yy}(s,t) \end{cases},$$

$$b'(s,t) = \begin{cases} L'_{xy}(s,t) & \text{for} \quad L_{yy}(s,t) < L_{xx}(s,t) \\ \lambda'_L(s,t) - L'_{xx}(s,t) & \text{for} \quad L_{xx}(s,t) < L_{yy}(s,t) \end{cases},$$

$$\lambda'_L(s,t) = \frac{1}{2}\left(L'_{xx} + L'_{yy} + \frac{L'_{xx}L_{xx} + L'_{yy}L_{yy} + 4L'_{xy}L_{xy} - L'_{xx}L_{yy} - L'_{yy}L_{xx}}{\sqrt{L_{xx}^2 + L_{yy}^2 + 4L_{xy}^2 - 2L_{xx}L_{yy}}}\right),$$

$$L'_{\alpha\beta}(s,t) = \frac{1}{\mathrm{Tr}\left(\boldsymbol{\sigma}(t)\right)}\frac{\partial f_{\alpha\beta}(s,s,t)}{\partial t} - \frac{\mathrm{Tr}\left(\boldsymbol{\sigma}'(t)\right)}{\left(\mathrm{Tr}\left(\boldsymbol{\sigma}(t)\right)\right)^2} f_{\alpha\beta}(s,s,t).$$

For the sake of its use as part of this new functional $h_3(\mathbf{f}_{\mathrm{Prin}}(s,s,t))$, note that $\mathbf{V}_{B(s)}(s,t)$ can be written entirely in terms of $L_{\alpha\beta}(s,t)$ and $L'_{\alpha\beta}(s,t)$ which in turn only depend on $f_{\alpha\beta}(s,s,t)$ and $\frac{\partial}{\partial t}f_{\alpha\beta}(s,s,t)$. $\mathbf{V}_{B(s)}(s,t)$ can be written as a function of $\mathbf{f}(s,s,t)$ so that $\mathbf{V}_{B(s)}(\mathbf{f}(s,s,t))$ and correspondingly $\left[\mathbf{V}_{B(s)}(s,t)\right]_{\mathrm{Prin}}$ can be written as $\mathbf{V}_{B(s)}(\mathbf{f}_{\mathrm{Prin}}(s,s,t))$. This feature makes $\mathbf{V}_{B(s)}$ suitable for use as part of a functional. Formulas for a, b, λ_L and $L_{\alpha\beta}$ for equation (4.8) were previously defined in equation (4.5). The derivation of equation (4.8) is very similar to that of $\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}t}$ in appendix C, and thus is omitted. In equation (4.6) the only dependence on s and t is in the form of $(\mathbf{f}_{\mathrm{Prin}}(s,s))_{\alpha\beta}$ and it can be rearranged to only depend on combinations of $(\mathbf{f}_{\mathrm{Prin}}(s,s))_{11} - (\mathbf{f}_{\mathrm{Prin}}(s,s))_{22}$ and $(\mathbf{f}_{\mathrm{Prin}}(s,s))_{12}$. This aligns with our motivation in section 4.3.1.

With the addition of a third functional to consider, the method established in section 4.1.3 remains largely the same, with one more Γ_i variable to track the evolution of using the Euler method. The main implication of changing to three functionals is that the functional phase space used during interpolation has one additional dimension, requiring more sophisticated interpolators.

4.4 Generalisation of the interpolator

The introduction of a third functional means that our phase space is now three-dimensional. The quadrant interpolator from appendix B.1.1 that we have used thus far for uniaxial extension is not suitable for interpolation in 3D and so revisions need to be made. In this section, we first discuss the effect a third functional has on the training set in section 4.4.1 and how to revise interpolators so they are applicable to this three-dimensional phase space in section 4.4.2, with two specific examples in appendices B.2.1 and B.2.2. The major complications to the creation of an interpolator in three-dimensional phase space and our proposed solutions are detailed in sections 4.4.3 and 4.4.4.

4.4.1 Production of a training set

Let us first review the production of a training set given two functionals under the rotation-based model in section 4.1.3 before we discuss alterations that need to be made for a three-functional case. We first run the GLaMM model at a selection of flow rates for chosen flow geometries, this will return the evolution of $\mathbf{f}_{\text{Lab}}^{\text{Train}}$ for each rate. From this, we use $\mathbf{f}_{\text{Prin}} = \mathbf{V}^T \cdot \mathbf{f}_{\text{Lab}} \cdot \mathbf{V}$ to produce the corresponding $\mathbf{f}_{\text{Prin}}^{\text{Train}}$ data, which always has principal direction in the $\hat{\mathbf{x}}$ direction by definition. We then calculate each $\Gamma_i^{\text{Train}} = h_i(\mathbf{f}_{\text{Prin}}^{\text{Train}})$ and record $\mathbf{f}_{\text{Prin}}^{\text{Train}}$, Γ_1^{Train} and Γ_2^{Train} for each flow at each timestep. A data point in this training set then consists of a known $(\mathbf{f}_{\text{Prin}}^{\text{Train}}(s, s'))_{\alpha\beta}$, defined for all α, β, s and s', along with its coordinates in functional phase space $(\Gamma_1^{\text{Train}}, \Gamma_2^{\text{Train}})$. These points are non-uniformly distributed throughout Γ_i phase space. For our results in sections 3 and 4.2 the training set we used consisted entirely of uniaxial extension data. Uniaxial extensional flows in

the x-direction and their corresponding relaxations have $\mathbf{f}_{\text{Lab}} \equiv \mathbf{f}_{\text{Prin}}$ and so this training set did not need alteration when we adopted the rotation revisions in section 4.1.3.

With three functionals, we have one more Γ_i^{Train} to calculate and record. The creation of the training set is largely the same, the only major difference being that with three functionals, the functional phase space is now three-dimensional rather than two. The uniaxial extension data was sufficient to span the physically realisable region when the phase space was two-dimensional, but this is not the case for a three-dimensional phase space. Using the definition in equation (4.6), $h_3(\mathbf{f}_{\text{Prin}}) = 0$ for uniaxial extension flows by design and so we require training data from other flow geometries if we hope to span this 3D phase space. Thus we calculate the evolution of xy shear flows of a selection of rates from $\dot{\gamma}\tau_R = 0.001$ to $\dot{\gamma}\tau_R = 10$ with the GLaMM model, rotate them into the principal reference frame, and include these in the training set. To evaluate the predictions under this new Γ_3 from equation (4.6), training data from these uniaxial extension flows and shear flows will be sufficient.

4.4.2 Interpolator generalisation for 3D

The interpolator also operates in fundamentally the same way as 2D. The aim of the interpolator in our model is to produce a $\mathbf{f}_{\text{Prin}}^{\text{Pred}}$ prediction at a known $(\Gamma_1^{\text{Pred}}, \Gamma_2^{\text{Pred}}, \Gamma_3^{\text{Pred}})$ location. If the choice of functionals is suitable to uphold our previous assumptions, all $\mathbf{f}_{\text{Prin}}$ configurations that arrive at the same location $(\Gamma_1, \Gamma_2, \Gamma_3)$ in phase space through evolutions, where $\Gamma_i^{\text{Train}} = h_i(\mathbf{f}_{\text{Prin}})$, will have similar $\mathbf{f}_{\text{Prin}}$ configurations regardless of flow history. Thus an $\mathbf{f}_{\text{Prin}}^{\text{Train}}$ associated with the point $(\Gamma_1^{\text{Train}}, \Gamma_2^{\text{Train}}, \Gamma_3^{\text{Train}})$ will be a good approximation to any $\mathbf{f}_{\text{Prin}}$ configuration with the same functional

values, and so our training data can be used for predictions. The better our assigned functionals are at uniquely characterising each potential $\mathbf{f}$ configuration, the more accurate this approximation is. An interpolator should calculate some weighted average based on the value of $\mathbf{f}_{\text{Prin}}^{\text{Train}}$ at positions close to $(\Gamma_1^{\text{Pred}}, \Gamma_2^{\text{Pred}}, \Gamma_3^{\text{Pred}})$ in the phase space. With a suitable choice of functionals, the closer a training point is in this phase space, the more relevant its corresponding $\mathbf{f}_{\text{Prin}}^{\text{Train}}$ is to prediction, and the heavier weighting it should be given.

The four-point quadrant interpolator that we have used for interpolation in two-dimensional phase space is no longer applicable in three-dimensional phase space applications, and so a different interpolator must be used. A natural choice is to generalise the quadrant interpolator which takes four points, one point from each quadrant, to an octant interpolator which interpolates using eight points from the eight octants in 3D space. Alternatively, we could choose some radius r around our prediction point in phase space and interpolate using all training points that exist within this radius. We refer to these interpolators as the octant interpolator and radial interpolator respectively and further details on their formulae can be found in appendices B.2.1 and B.2.2.

4.4.3 Absolute and relative interpolation scaling

We make one further change to these interpolators to improve their performance when predicting shear flows. Thus far we have used absolute scaling in each of our interpolators. We normalise the size of results from each of our functionals by taking the maximum value that each Γ_i^{Train} individually attains within our training set and scaling all distances by this value. This

scaling is written as,

$$M_i = \max(\Gamma_i^{\text{Train}}) \in \mathbf{\Omega}, \tag{4.9}$$

and is used in the quadrant, octant and radial interpolators in equations B.1, B.4 and B.5 respectively. When scaled in this way, each dimension of the functional phase space is bounded so that $\mathbf{\Omega} \subset [0,1]^3$.

Issues can occur when the maximums that Γ_i^{Test} attains during evolution with the GLaMM model significantly differ from the maximums of Γ_i^{Train}. For example, let us consider a shear flow with our current choice of functionals and training set from sections 4.3.2 and 4.4.1. $\max(\Gamma_3^{\text{Test}})$ may or may not be similar to $\max(\Gamma_3^{\text{Train}})$ depending on the chosen shear rate, but $\max(\Gamma_1^{\text{Test}}) \ll \max(\Gamma_1^{\text{Train}})$ and $\max(\Gamma_2^{\text{Test}}) \ll \max(\Gamma_2^{\text{Train}})$ are always true because our training set includes uniaxial extension flows which have values of Γ_1 and Γ_2 that are far larger than anything that can be attained by shear flows. In this situation, interpolator performance may be improved if the scaling was tweaked so that the interpolator prioritises similar values for Γ_1^{Train} and Γ_2^{Train} and allows Γ_3^{Train} to vary more. The current absolute scaling in equation (4.9) weights interpolator distance in each direction equivalently, and so we introduce a relative scaling,

$$M_i^R(\Gamma_i^{\text{Pred}}) = \max(\Gamma_i^{\text{Pred}}, 10^{-10} M_i).$$

This relative scaling in equation (4.4.3) allows the interpolator to adapt the priority of each direction of phase space based on the current prediction location, it is not fixed at a set value based on the training set. For example, in a situation where Γ_3^{Pred} is twice the value of the Γ_1^{Pred} and Γ_2^{Pred}, interpolation distance will be halved in the Γ_3 direction for the purposes of point selection and weighting. This will help the interpolator make more sensible choices when predicting polymer flows that remain within a small

region of $\mathbf{\Omega}$ in functional phase space during their evolution.

4.4.4 Interpolation bias

Due to the non-uniform distribution of our training data throughout phase space, these interpolators can suffer from bias. Depending on the choice of points the weighted average location in functional phase space may differ from the location $(\Gamma_1^{\text{Pred}}, \Gamma_2^{\text{Pred}}, \Gamma_3^{\text{Pred}})$ at which the interpolation was performed. The effects of interpolation bias are more pronounced in regions of lower density within the training data. With two functionals the effects of interpolation bias are small since we defined a sufficiently dense training data set that spanned all realistic combinations of Γ_1 and Γ_2, but sparse regions are more common in three-dimensional phase space when using an additional functional.

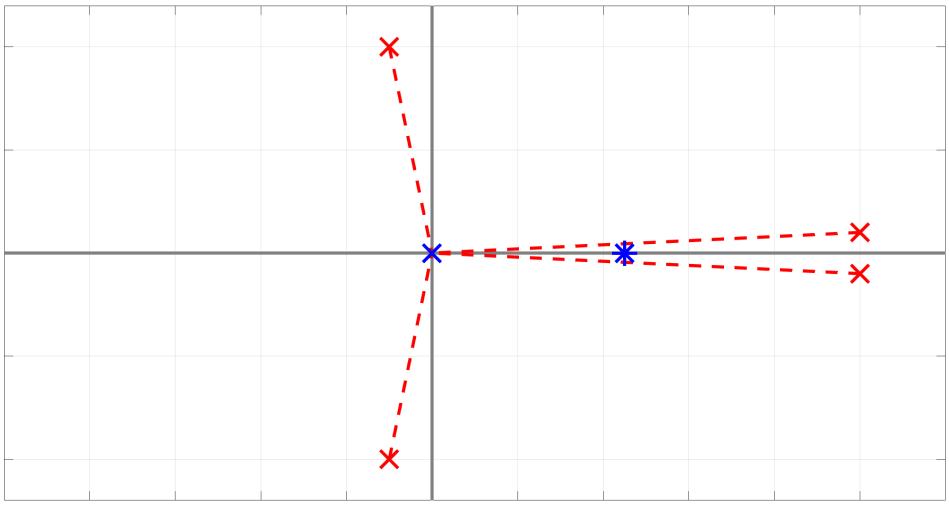


Figure 4.5: A visualisation of how the 2D quadrant interpolator might suffer from bias. The location to be predicted is the blue cross, the red crosses represent the training points chosen by the quadrant interpolator, and the blue star shows their average location. The octant interpolator can have similar issues in 3D.

As figure 4.5 shows, when choosing a number of closest points, the weighted average location of $(\Gamma_i^{\text{Train}})_q$ can differ from Γ_i^{Pred}. The resulting $\mathbf{f}_{\text{Prin}}^{\text{Pred}}$ from this interpolation would have $h_i(\mathbf{f}_{\text{Prin}}^{\text{Pred}}) \neq \Gamma_i^{\text{Pred}}$. This bias effect is more pronounced with distant points, thus why sparse regions are at higher risk. These same features are present in the octant interpolator when we extend the functional phase space to three-dimensional. The radial interpolator is slightly more robust to bias in sparse regions of the training data due to the increased number of points considered, but as it does not restrict the number of points in each octant, is it more vulnerable to bias in regions of training data with non-uniform density.

If interpolation bias becomes a problem for a particular training data set, the simplest solution is to add more training data in the problematic regions. An increase in the amount of training data does increase the computation time of the interpolator, but thinning the number of points in the densest regions of the training data gives control over the compromise between interpolation bias and speed. If removing interpolation bias by adding more training data is not feasible, another option is to redefine or transform your functionals, h_i. The functionals define the location of each training data point in functional phase space, and a functional where the size of the change in the value of Γ_i is proportional to the size of significant change will result in a more uniform density for the training set and fewer sparse regions overall. A further option to prevent interpolation bias would be to create a new interpolation scheme that is more robust to bias.

We have also explored a method to remove interpolation bias by transforming the resultant $\mathbf{f}_{\text{Prin}}$ from interpolation, dubbing this transformed $\mathbf{f}_{\text{Prin}}^{*}$ as $\mathbf{f}_{\text{Prin}}^{\text{Pred}}$. By using the correct transformations we ensured that $h_i(\mathbf{f}^{*}) = \Gamma_i^{\text{Pred}}$, thereby removing interpolation bias. Unfortunately, we found that these transformations had a detrimental impact on $\frac{\partial}{\partial t}\mathbf{f}_{\text{Prin}}^{\text{Pred}}$ and that the evolution

predictions with the model from section 4.1.3 worsened overall due to the impact on $\frac{\mathrm{d}\Gamma_i}{\mathrm{d}t}$.

4.4.5 Guidance on composition of a suitable training set

The prediction accuracy of our model is heavily dependent on the creation of a suitable training set. The suitability of a certain selection of flow rates as a training set is intrinsically linked to the functionals we choose, a different set of functionals would change how the training data spans the functional phase space $\mathbf{\Gamma}$. For this reason, it is important that we outline the properties to be considered for a training set to be suitable for use with our reduced model and how to create a training set that meets these requirements. In general, the aim when building a training set for our reduced model is to ensure that training data points sufficiently span the functional phase space so that our interpolations are accurate without requiring a large amount of data. With the octant interpolator choice, one of the most important factors is to ensure that each point you wish to interpolate is surrounded by at least one point in each octant. If the interpolator is not able to use a point from each octant, this leads to extrapolation of the data, significantly reducing accuracy. In addition, the octant interpolator performs better if the data are sufficiently dense that the interpolation bias is low. A sparse training set can also introduce noise into our predictions with certain interpolators, as will be seen throughout this chapter. The "Radial octant average" interpolator we introduce later in section 4.7 also has these same requirements, but is far more resilient to sparsity induced noise, due to its construction.

To meet these requirements without requiring a vast amount of GLaMM

model simulations, there are certain techniques we can employ. As the underlying GLaMM model has physical implications and limitations, for a suitable choice of functionals, it tends to be the case that not every location in $\mathbf{\Gamma}$ space is reachable, as some combinations of functionals may not have an associated $\mathbf{f}(s, s')$ configuration that is physically realistic. Instead, training flow trajectories tend to span a specific region of $\mathbf{\Gamma}$ space. An example of this for the 2D case can be seen in figure 3.6. For the creation of a new training set, we first simulate a small number of flows to get an idea of the span of the set, making sure to include the trajectories for the highest and lowest rates of interest.

Once we have identified the span of the trajectories we add new trajectories with two goals in mind. The first objective is to try to get a number of points on the edges of this span, including any trajectories that would introduce training points on this edge. Having points on the edge of the span ensures that the interpolator will not need to extrapolate when close to that edge. The second objective is to ensure there are no sparse regions within the span of the training data, adding simulated trajectories as necessary. Provided your choice of functionals is suitable, a denser training set will improve interpolation accuracy but will incur a slight computational cost. The overall aim is to ensure that your interpolator has a sufficient number of nearby points in as large a region as possible, minimising the locations where it will need to extrapolate or could suffer from bias. An interpolator that does not suffer from extrapolation or bias will be far more robust when predicting various flows.

The calculation of a large number of GLaMM model evaluations to create the training set can take a significant amount of time, and if you were to create an entire training set to evaluate a single polymer flow with our reduced model, it would not be productive. The advantages become ap-

parent because the training set is a one-time cost, once a training set is created it can be saved and, the reduced model can be used any number of times on any flow that is within the span of the training set, including qualitatively different ones such as variable rate flows. Over a number of repeated runs, the vastly increased speed of our reduced model easily outweighs this initial cost. The interpolation does slightly increase in cost with a larger number of points in the training set. If the computation cost of the training set becomes a problem, the density of the training set could be reduced to improve efficiency. After simulation, the training points in $\mathbf{\Gamma}$ phase space will lie on a non-uniform grid, varying density in different regions. A possible way to reduce density would be to impose a uniform grid of points, only including the single closest training point to each point on the grid, thereby making the sparsity of the training set much closer to uniform. However, in our calculations, the evaluation of the training set with the interpolator was always a small amount of total cost, less than 8%, and so it was less necessary to take steps to reduce the density.

We also tested if the inclusion of uniaxial extension data in the training set was necessary when predicting shear flows. We found that some shear flow trajectories that were completely surrounded by training data trajectories were still able to be predicted, but that many others suffered greatly. One of the major reasons for this is that the uniaxial extension training data is on the edge of the span of training data, and the removal of these points leads to far more extrapolations during prediction. In addition, our reduced model was far less robust without uniaxial extension data. For variable rates such as cessation of shear, Γ_3 drops to zero quickly, and thus the uniaxial extension data with $\Gamma_3 = 0$ makes interpolation of such flows possible. The loss of robustness on exclusion and low interpolation cost led us to make the decision that our reduced model is best with a single

training set that includes both extension and shear data.

Our results later in this chapter show that our choices for the training set in section 4.4.1 are reasonable and can predict a wide range of flows accurately. Some of our early results suffer from noise, which is a sign that the training data may be too sparse for the octant interpolator, but we conclude that this is due to the sensitivity of the octant interpolator, rectifying this with a new interpolator in section 4.7.

4.5 Initial results with three functionals

Now that we have a revised method from section 4.1.3, three functionals defined in equations (4.3), (4.4) and (4.6), training data of both uniaxial extension and shear flows from section 4.4.1 and two interpolators for 3D functional phase space in equations (B.4) and (B.5), we are in a position to predict shear flow using three functionals. For this section, we show how our model performs in predicting a shear flow of rate $\dot{\gamma}\tau_R = 9$ with CCR on $C_\nu = 0.1$. All other GLaMM model parameters are defined in section 2.3.

As we have highlighted interpolation bias as a more significant problem with the increased sparsity of a training set in 3D, we first evaluate interpolator bias for both interpolators with the relative scaling introduced in equation (4.4.3). To do this we run the GLaMM model for our chosen shear flow and replace the predicted functional values with the values from the GLaMM model $\Gamma_i^{\text{Pred}} = \Gamma_i^{\text{Test}}$ at every single timestep. This means that we forgo our reduced model's step of calculating the evolution with the Euler method using $\frac{\mathrm{d}}{\mathrm{d}t}\Gamma_i^{\text{Pred}}$ and removes any possible accumulation of errors from this calculation. We note that whenever we require Γ_i^{Test} or Γ_i^{Train} we calculate

them directly using our functionals so that $h_i(\mathbf{f}^{\text{Test}}_{\text{Prin}}) = \Gamma_i^{\text{Test}}$ and $h_i(\mathbf{f}^{\text{Train}}_{\text{Prin}}) = \Gamma_i^{\text{Train}}$. This is not the same relationship for predicted variables however, we receive Γ_i^{Pred} from the previous timestep and we use the interpolator to recover a prediction so that $g(\Gamma_i^{\text{Pred}}) = \mathbf{f}^{\text{Pred}}_{\text{Prin}}$. Any interpolation bias or other interpolator errors will prevent equality, explicitly $h_i(\mathbf{f}^{\text{Pred}}_{\text{Prin}}) = \Gamma_i^{\text{Pred}} + \epsilon_i$. Thus if we compare the interpolator input, Γ_i^{Pred}, to the functional phase space location of the interpolator output, $h_i(\mathbf{f}^{\text{Pred}}_{\text{Prin}})$, we can isolate these interpolation errors. For this chosen shear rate, the interpolation error for the octant and radial interpolator can be seen in the figures 4.6 and 4.7 respectively.

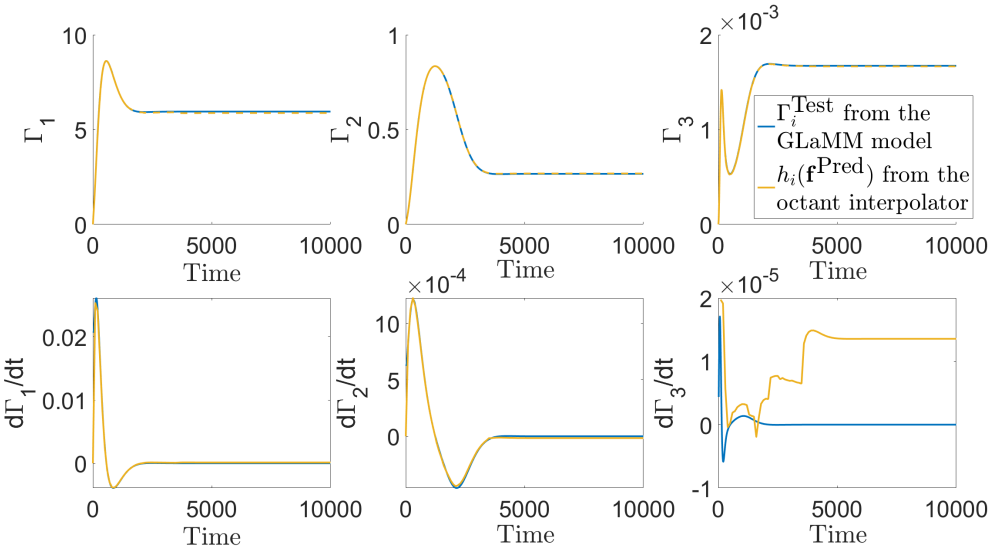


Figure 4.6: A plot to show the interpolation error in Γ_i and the consequent errors in derivatives under the new reduced method with the octant interpolator and relative scaling. The dashed line indicates when the octant interpolator uses less than 8 points. Test data is a shear flow of rate $\dot{\gamma}\tau_R = 9$. Training data is a mix of uniaxial extension flows and shear flows.

Figures 4.6 and 4.7 show that both of these interpolation methods are capable of producing a $\mathbf{f}^{\text{Pred}}_{\text{Prin}}$ with functional values close to the interpolation location, Γ_i^{Pred}. We can reduce the density of the training data and retain an acceptable level of interpolator bias. Through testing, the closest rates we require to be in the training set for a reasonable degree of accuracy is $\dot{\gamma}\tau_R =$

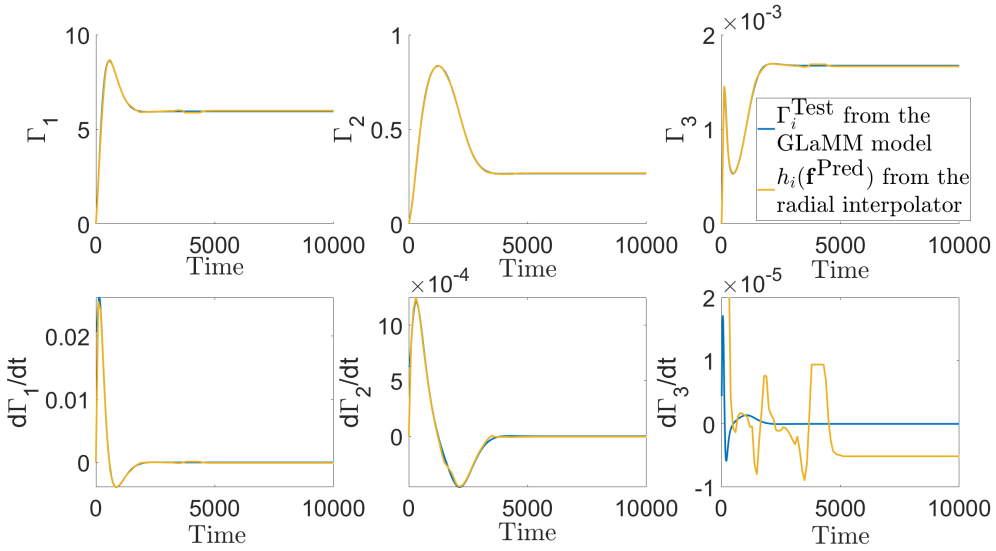


Figure 4.7: A plot to show the interpolation error in Γ_i and the consequent errors in derivatives under the new reduced method with the radial interpolator and relative scaling. Test data is a shear flow of rate $\dot{\gamma}\tau_R = 9$. Training data is a mix of uniaxial extension flows and shear flows.

9 ± 0.5. The $\frac{\mathrm{d}}{\mathrm{d}t}\Gamma_i^{\mathrm{Pred}}$ derivatives in figures 4.6 and 4.7 are calculated using our new method from section 4.1.3, but we do not iterate these results forward. The true Γ_i^{Test} is provided at each timestep in order to analyse errors in interpolation without the accumulation of errors over time. We can see that both interpolation methods give good predictions for $\frac{\mathrm{d}\Gamma_1}{\mathrm{d}t}$ and $\frac{\mathrm{d}\Gamma_2}{\mathrm{d}t}$ but struggle to correctly capture $\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t}$. This suggests that whilst the basic choice of h_3 from equation (4.6) would appear to be a suitable choice, its derivative is very sensitive to the errors in $\mathbf{f}_{\mathrm{Prin}}^{\mathrm{Pred}}$ caused by these interpolation methods. This is one of the main reasons that prompts a later revision to the choice of h_3 in equation (4.10).

Before we alter the choices of any functionals, let us first see how our new model with three functionals performs at predicting the evolution of a shear flow with rate $\dot{\gamma}\tau_R = 9$. From this point forward, we choose to use the octant interpolator. Results for the radial interpolator are similar, but overall the octant interpolator produces better results.

Our investigation into predicting uniaxial extension flows in chapter 3 shows that interpolation loses accuracy when all of $\mathbf{f}$ is close to isotropic. Furthermore, the calculation of $\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}t}$ is problematic when $\mathbf{f}$ is isotropic. However, the problem only arises at very low strain which is most common at the start of a flow from a fully relaxed fluid. Herein, we bypass this by beginning a short period into the flow and using the true GLaMM data as an initial condition. For all following calculations we choose this period to be $\frac{200}{\tau_e}$. This approach is suitable for our predictions, but a more complete solution to this issue could be provided by switching the model when the deformation is small. Multiple different models would work well for this region, suitable choices include the Rolie-Poly model, a linearised form of the GLaMM model, or even the full GLaMM model. We discuss these possibilities in more detail in section 5.2. Thus we employ this approach to produce figures 4.8, 4.9 and 4.10. These figures show the predicted evolution of the Γ_i, $\mathbf{S}$ tensor and some common stress combinations respectively. Accuracy in Γ_i^{Pred} and $\mathbf{S}^{\mathrm{Pred}}$ prove model consistency, whilst the final graph shows how our model performs in predicting physical stress quantities, with comparisons to the full GLaMM model, and a current model reduction, the Rolie-Poly model.

Throughout this chapter, the colour of the lines will help identify the current frame of reference for the calculation. Blue and orange lines represent calculations in the principal frame that use $\mathbf{f}_{\mathrm{Prin}}^{\mathrm{Test}}$ and $\mathbf{f}_{\mathrm{Prin}}^{\mathrm{Pred}}$ respectively, whilst green, red and purple are reserved for laboratory frame calculations and signify the use of $\mathbf{f}_{\mathrm{Lab}}^{\mathrm{Test}}$, $\mathbf{f}_{\mathrm{Lab}}^{\mathrm{Pred}}$ or the Rolie-Poly model respectively.

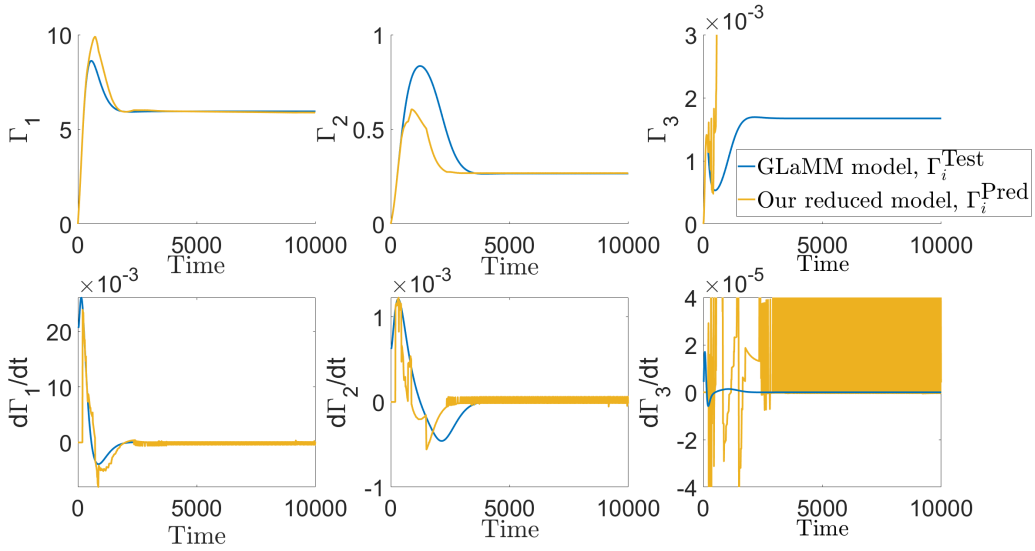


Figure 4.8: Γ_i predictions from the new reduced model compared to the full GLaMM model solutions, for a shear flow of rate $\dot{\gamma}\tau_R = 9$.

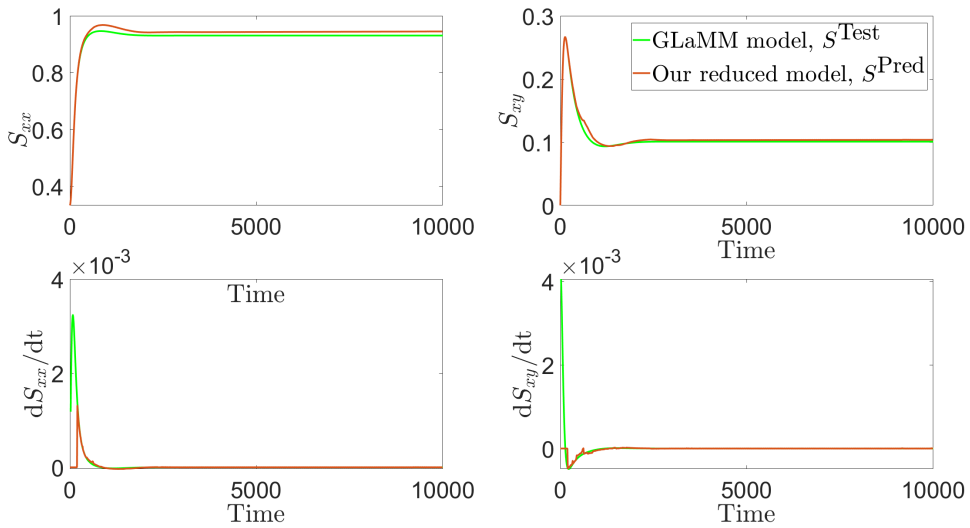


Figure 4.9: **S** tensor predictions from the new reduced model compared to the full GLaMM model solutions, for a shear flow of rate $\dot{\gamma}\tau_R = 9$.

Figure 4.8 suggests that there are significant errors in the prediction of Γ_i, particularly Γ_3. Despite this, these functionals, h_i, are able to capture the tensor for principal direction, **S**, well, as evident from figure 4.9. Furthermore, figure 4.10 indicates an improvement in the predictions of the shear stress and first normal stress difference compared to the two-functional method from section 4.2, but numerical noise is present in these predictions with three functionals. This noise can be explained by looking at the

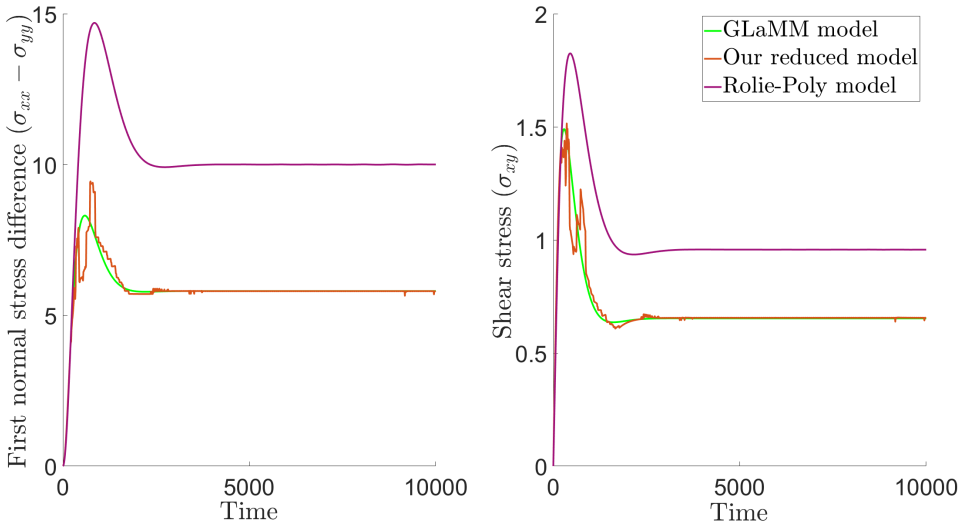


Figure 4.10: Stress predictions from the new reduced model compared to the full GLaMM model solutions and the Rolie-Poly model solutions, for a shear flow of rate $\dot{\gamma}\tau_R = 9$. The timestep used in the new reduced model is 900 times larger than the GLaMM model with a start time of $t = 200$.

graph of Γ_3. The inaccuracy in Γ_3 causes noise in each $\frac{\mathrm{d}}{\mathrm{d}t}\Gamma_i^{\mathrm{Pred}}$, which in turn causes numerical noise to be present in the stress quantities. Once Γ_3^{Pred} incurs errors sufficient to cause its value outside of the region that is well defined in our training set, both of our current 3D interpolators encounter significant difficulty in producing a $\mathbf{f}_{\mathrm{Prin}}^{\mathrm{Pred}}$ at a location that is beyond the scope of the training set. These interpolation errors manifest as numerical noise in the values of $\frac{\mathrm{d}}{\mathrm{d}t}\Gamma_i^{\mathrm{Pred}}$. We can prove this by replacing the reduced model's predictions, Γ_3^{Pred}, with the corresponding value directly from the GLaMM test data, Γ_3^{Test}, and performing the new reduced model method for all other components. Doing so produces figure 4.11.

Figure 4.11 shows the reduction in error in Γ_i^{Pred} and $\frac{\mathrm{d}}{\mathrm{d}t}\Gamma_i^{\mathrm{Pred}}$ when Γ_3 is fixed, and if we produce predictions for stress under these conditions, we get a similar reduction in noise. And so removing the accumulation of errors in Γ_3 fixes all errors except the noise in $\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t}$. This noise remains present for the same reason it is present in figure 4.6, $\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t}$ is very sensitive to errors in $\mathbf{f}_{\mathrm{Prin}}^{\mathrm{Pred}}$ from the octant interpolator.

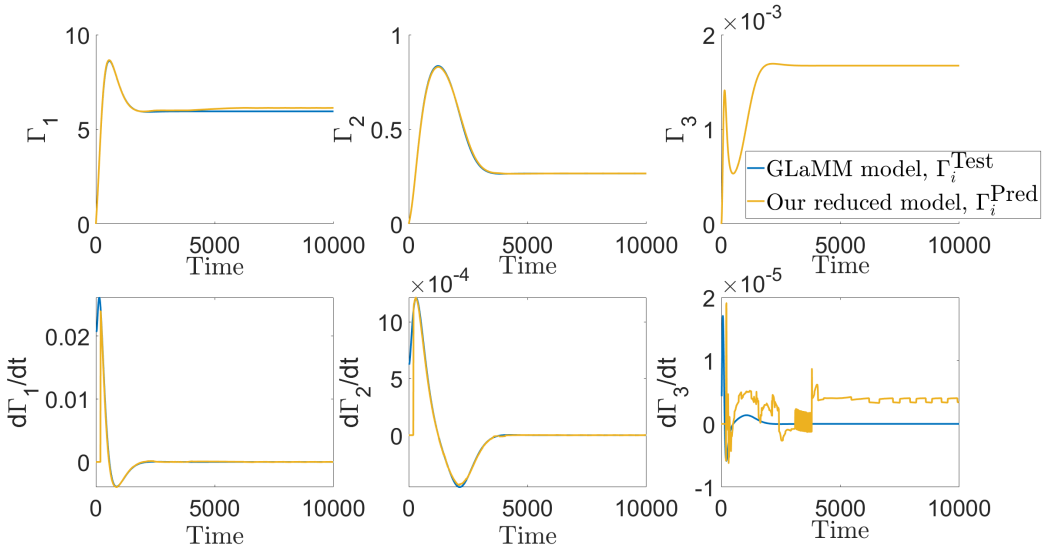


Figure 4.11: Γ_i predictions from the new reduced model compared to the full GLaMM model solutions, for a shear flow of rate $\dot{\gamma}\tau_R = 9$. Γ_3^{Pred} is replaced by Γ_3^{Test}.

Even with this level of noise, our new reduced model is better at predicting shear stress and the first normal stress difference than the ad-hoc model reduction of the Rolie-Poly model, with a timestep that is 900 times larger than the full GLaMM model but does suffer from numerical noise. It is also an improvement on the shear flow predictions that used two functionals in section 4.2. The remaining errors all appear to stem from our choice of h_3 in equation (4.6) and its interaction with the interpolator, which suggests that a different choice could improve prediction accuracy.

4.6 Establishing an improved third functional

4.6.1 Issues with the initial three-functional choice

We have shown our reduced model can predict shear flows well using three functionals, despite the prediction being subject to numerical noise. Small errors in the $\mathbf{f}_{\text{Prin}}^{\text{Pred}}$ from the interpolator significantly influence $\frac{\mathrm{d}}{\mathrm{d}t}\Gamma_3^{\text{Pred}}$ pre-

dictions, this in turn causes Γ_3^{Pred} to evolve wrongly, inducing numerical noise in the predictions of $\frac{\mathrm{d}}{\mathrm{d}t}\Gamma_1^{\text{Pred}}$ and $\frac{\mathrm{d}}{\mathrm{d}t}\Gamma_2^{\text{Pred}}$, and subsequently in elements of our stress predictions, $\boldsymbol{\sigma}^{\text{Pred}}$. To find the source of this noise, we turn our attention to the derivatives of each functional $\frac{\mathrm{d}\Gamma_i}{\mathrm{d}t}$. Our method categorises each $\mathbf{f}_{\text{Prin}}^{\text{Train}}$ in our training set by calculating the value of the three Γ_i functionals, and so, two training $\mathbf{f}_{\text{Prin}}^{\text{Train}}$ configurations that result in similar Γ_i will be similar configurations. Thus, the distance between two training points in Γ_i phase space is proportional to the similarity between their respective $\mathbf{f}_{\text{Prin}}^{\text{Train}}$ configurations by design. However, under our current formulation $\frac{\mathrm{d}\Gamma_i}{\mathrm{d}t}$ does not always conform to the same proportionality. This can cause issues, especially when the interpolator chooses a training point that is at a different point in shear evolution where the difference in $\frac{\mathrm{d}\Gamma_i}{\mathrm{d}t}$ is far larger than the phase space distance would suggest.

One area where this issue is prevalent is at early times, where significant non-monotonicity in Γ_i causes many points that are at different points in shear evolution to have similar Γ_i values. This increases the likelihood that the interpolator will use $\mathbf{f}_{\text{Prin}}^{\text{Train}}$ configurations from different phases of relaxation which may have significantly different $\frac{\mathrm{d}\Gamma_i}{\mathrm{d}t}$ values to what the true configuration has. Another area where this issue can cause significant problems is steady state. In steady state, it is less likely that all eight octants of phase space will contain a training configuration that is close to the prediction location, leading to the use of distant configurations from other stages in their respective relaxations. Whilst the overall effect on $\mathbf{f}_{\text{Prin}}^{\text{Train}}$ and Γ_i^{Pred} is tempered by our distance weighting within the interpolator this is not necessarily the case for $\frac{\mathrm{d}}{\mathrm{d}t}\Gamma_i^{\text{Pred}}$. When calculating the average $\mathbf{f}_{\text{Prin}}^{\text{Pred}}$ using the interpolator, it is possible to inherit some feature from this 'distant' configuration that even when scaled down in accordance to our interpolator's distance weighting formula has a significant effect on the resulting

$\frac{\mathrm{d}}{\mathrm{d}t}\Gamma_i^{\mathrm{Pred}}$ despite this same feature being insignificant in the calculation of Γ_i^{Pred} at this reduced size. These effects can be seen in figures 4.8 and 4.10 where the noise is more significant at the early times and in steady state.

4.6.2 Interpolation visualisation

As an example, consider this interpolation snapshot, figure 4.12.

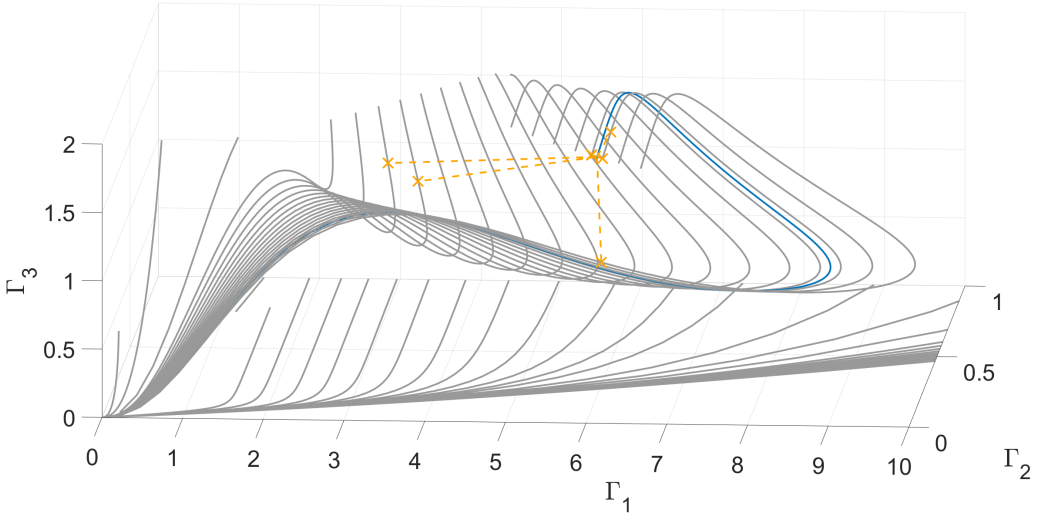


Figure 4.12: Point choice for the octant interpolator under relative scaling in functional phase space. The $\mathbf{f}_{\mathrm{Prin}}^{\mathrm{Test}}$ to be predicted is from a shear flow of rate $\dot{\gamma}\tau_R = 9$ in its steady state. This test data is depicted in blue and the point choice from the grey training set is shown in orange. Note that the interpolator only found relevant points in seven octants on this occasion.

This snapshot in figure 4.12 shows an interpolation performed during the calculation of the Γ_i^{Pred} and $\frac{\mathrm{d}}{\mathrm{d}t}\Gamma_i^{\mathrm{Pred}}$ predictions for figure 4.6. Since Γ_i^{Pred} is relatively accurate in figure 4.6, this suggests that the points used for interpolation are sufficiently close in functional phase space that interpolation bias during functional estimation is not an issue, but this is not necessarily the case for their derivatives.

For example, let us use the definition of the octant interpolator from section B.2.1 to predict the test data at the location $\mathbf{\Gamma}^{\mathrm{Test}} = (\Gamma_1^{\mathrm{Test}}, \Gamma_2^{\mathrm{Test}}, \Gamma_3^{\mathrm{Test}})$

in functional phase space. Let us assume that all eight points are sufficiently close that interpolation bias is small, so that $h_3(\mathbf{f}_{\text{Prin}}^{\text{Pred}}) \approx \Gamma_3^{\text{Test}}$, but note that this does not imply that to $\frac{\mathrm{d}}{\mathrm{d}t}h_3(\mathbf{f}_{\text{Prin}}^{\text{Pred}}) \approx \frac{\mathrm{d}}{\mathrm{d}t}\Gamma_3^{\text{Test}}$. This is because the locations, $(\Gamma_1, \Gamma_2, \Gamma_3)$, of two associated points in functional phase space and their $h_i(\mathbf{f}_{\text{Prin}})$ values are directly linked by definition. A distance D between two points in functional phase space is equivalent to a difference of $\sqrt{\sum_{i=1}^{3}(h_i((\mathbf{f}_{\text{Prin}})_1) - h_i((\mathbf{f}_{\text{Prin}})_2))^2} = D$ in functional values, but no such restriction a relation between distance and difference in $\frac{\mathrm{d}}{\mathrm{d}t}h_i(\mathbf{f}_{\text{Prin}})$ exists. This can cause issues when $W_q(\mathbf{\Gamma}^{\text{Test}}) = 1/D$ but $\sqrt{\sum_{i=1}^{3}(\frac{\mathrm{d}}{\mathrm{d}t}h_i(\mathbf{f}_{\text{Prin}}^{\text{Test}}) - \frac{\mathrm{d}}{\mathrm{d}t}h_i((\mathbf{f}_{\text{Prin}}^{\text{Train}})_q))^2} \gg D$ as the weighting is not sufficient to suppress the size of the contribution to the interpolated $\mathbf{f}_{\text{Prin}}^{\text{Pred}}$.

This brings us back to the error in $\frac{\mathrm{d}}{\mathrm{d}t}\Gamma_3^{\text{Pred}}$ for steady state, depicted across two figures, figure 4.6 and figure 4.12. If only two close points are considered, $\frac{\mathrm{d}}{\mathrm{d}t}\Gamma_3^{\text{Pred}} \approx 0$. However, with octant interpolation, the two points to the left of figure 4.12 have a weighting that is approximately forty times smaller than the closest interpolation points, but the size of $\frac{\mathrm{d}}{\mathrm{d}t}h_3((\mathbf{f}_{\text{Prin}}^{\text{Train}})_q)$ at these points is two magnitudes of order larger, and thus their contribution to $\frac{\mathrm{d}}{\mathrm{d}t}h_3(\mathbf{f}_{\text{Prin}}^{\text{Pred}})$ eclipses all other points.

Possible ways to alleviate these issues include improving the choice of h_3 or creating a more sophisticated interpolator, both of which will be explored in this thesis. Whilst a monotonic h_3 is not required for this method, a functional that is closer to monotonic would reduce the number of times that similar Γ_3 values can have significant differences in $\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t}$, and thus reduce the noise at early times. In addition, a choice of h_3 with a derivative that is less sensitive to small fluctuations in $\mathbf{f}_{\text{Prin}}^{\text{Pred}}$ would reduce the effect of rogue features from distant configurations. Improving the interpolator could also weight these distant points more suitably or further limit their inclusion. We shall use both of these approaches in combination to reduce

noise, beginning with a revision to h_3.

4.6.3 Functional product combination

To successfully improve h_3 we should aim to increase the monotonicity of Γ_3 during a shear flow evolution, whilst maintaining the current accuracy with which we capture the stress tensor $\boldsymbol{\sigma}$. Since the prediction in figure 4.10 would be considered very accurate if the noise was absent, we continue to use the idea of variation in principal direction as a suitable functional for shear flows but modify the formula to be more monotonic. One way to alter this functional without changing the property it represents is to take a product between the current functionals. This way, the monotonicity of h_3 can be improved, and the variation in principal direction can be extracted by a combination of the new functionals so that no information is lost. Multiplying Γ_3 by Γ_1 or Γ_2 may not be sufficient to achieve an acceptable level of monotonicity and so we also explore product combinations of higher powers. The general formula for this revised Γ_3 is,

$$\Gamma_3 = h_3(\mathbf{f}_{\text{Prin}}) = h_{NSD}^{\alpha}(\mathbf{f}_{\text{Prin}})h_{OD}^{\beta}(\mathbf{f}_{\text{Prin}})h_{Var}(\mathbf{f}_{\text{Prin}}), \tag{4.10}$$

$$\begin{aligned}\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t} = \quad & \alpha\frac{\mathrm{d}h_{NSD}(\mathbf{f}_{\text{Prin}})}{\mathrm{d}t}h_{NSD}^{\alpha-1}(\mathbf{f}_{\text{Prin}})h_{OD}^{\beta}(\mathbf{f}_{\text{Prin}})h_{Var}(\mathbf{f}_{\text{Prin}}) + \\ & \beta\frac{\mathrm{d}h_{OD}(\mathbf{f}_{\text{Prin}})}{\mathrm{d}t}h_{NSD}^{\alpha}(\mathbf{f}_{\text{Prin}})h_{OD}^{\beta-1}(\mathbf{f}_{\text{Prin}})h_{Var}(\mathbf{f}_{\text{Prin}}) + \\ & \frac{\mathrm{d}h_{Var}(\mathbf{f}_{\text{Prin}})}{\mathrm{d}t}h_{NSD}^{\alpha}(\mathbf{f}_{\text{Prin}})h_{OD}^{\beta}(\mathbf{f}_{\text{Prin}}),\end{aligned} \tag{4.11}$$

where h_{NSD}, h_{OD} and h_{Var} are functionals that are defined as,

$$h_{NSD}(\mathbf{f}_{\text{Prin}}) = \frac{12G_e}{5}\int_0^Z (\mathbf{f}_{\text{Prin}}(s,s,t))_{11} - (\mathbf{f}_{\text{Prin}}(s,s,t))_{22}\; ds,$$

$$h_{OD}(\mathbf{f}_{\text{Prin}}) = \frac{12G_e}{5}\int_0^Z (\mathbf{f}_{\text{Prin}}(s, Z-s, t))_{11} - (\mathbf{f}_{\text{Prin}}(s, Z-s, t))_{22}\ ds,$$

$$h_{Var}(\mathbf{f}_{\text{Prin}}) = \int_0^Z 1 - (\hat{\mathbf{x}} \cdot \mathbf{V}_{B(s)}(\mathbf{f}_{\text{Prin}}(s, s, t)))^2\ ds,$$

$$\frac{\mathrm{d}h_{NSD}(\mathbf{f}_{\text{Prin}})}{\mathrm{d}t} = \frac{12G_e}{5}\int_0^Z \left(\frac{\partial}{\partial t}\mathbf{f}_{\text{Prin}}(s, s, t)\right)_{11} - \left(\frac{\partial}{\partial t}\mathbf{f}_{\text{Prin}}(s, s, t)\right)_{22} ds,$$

$$\frac{\mathrm{d}h_{OD}(\mathbf{f}_{\text{Prin}})}{\mathrm{d}t} = \frac{12G_e}{5}\int_0^Z \left(\frac{\partial}{\partial t}\mathbf{f}_{\text{Prin}}(s, Z-s, t)\right)_{11} - \left(\frac{\partial}{\partial t}\mathbf{f}_{\text{Prin}}(s, Z-s, t)\right)_{22} ds,$$

$$\frac{\mathrm{d}h_{Var}(\mathbf{f}_{\text{Prin}})}{\mathrm{d}t} = \int_0^Z -2(\hat{\mathbf{x}} \cdot \mathbf{V}_{B(s)}(\mathbf{f}_{\text{Prin}}(s, s)))\left(\hat{\mathbf{x}} \cdot \frac{\partial \mathbf{V}_{B(s)}(\mathbf{f}_{\text{Prin}}(s, s))}{\partial t}\right) ds,$$

The formula for $\mathbf{V}$ and $\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}t}$ remain the same as defined in equations (C.4) and (C.5). Our cursory investigation found that a choice of $\alpha = 1$ and $\beta = 3$ was sufficient to suppress the non-monotonic behaviour and that further increasing the powers had no significant effect on the results. A more in-depth exploration could find that a different choice of α and β results in a better final prediction, but given that $\alpha = 1, \beta = 3$ achieved our desired effect, and that the other values we evaluated had similar accuracy, we left this as a task for future work. We show the improvements when this combination is used in figures 4.13, 4.14 and 4.15.

Figures 4.13-4.15 show that the revised choice for h_3 shows significant improvements in the predictions of the evolutions of Γ_1^{Pred}, Γ_2^{Pred} and stress tensor components. With this combination, the model is now able to predict the components of the stress tensor in figure 4.15 to a sufficient level of accuracy, but the noise is not yet at an acceptable level. The noise is most prevalent in the functional derivatives in figure 4.13, where this problem is most pronounced when the steady state is reached. Thus, this area is where we focus our efforts when we explore other avenues to reduce this noise.

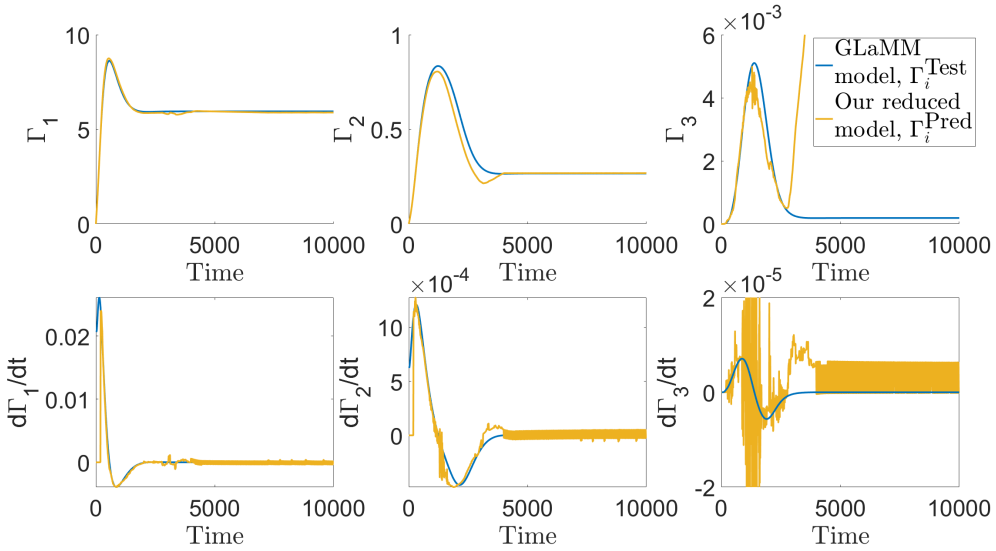


Figure 4.13: Γ_i predictions from the new reduced model compared to the full GLaMM model solutions, for a shear flow of rate $\dot{\gamma}\tau_R = 9$. h_3 is the revised choice from equation (4.10) with $\alpha = 1$ and $\beta = 3$.

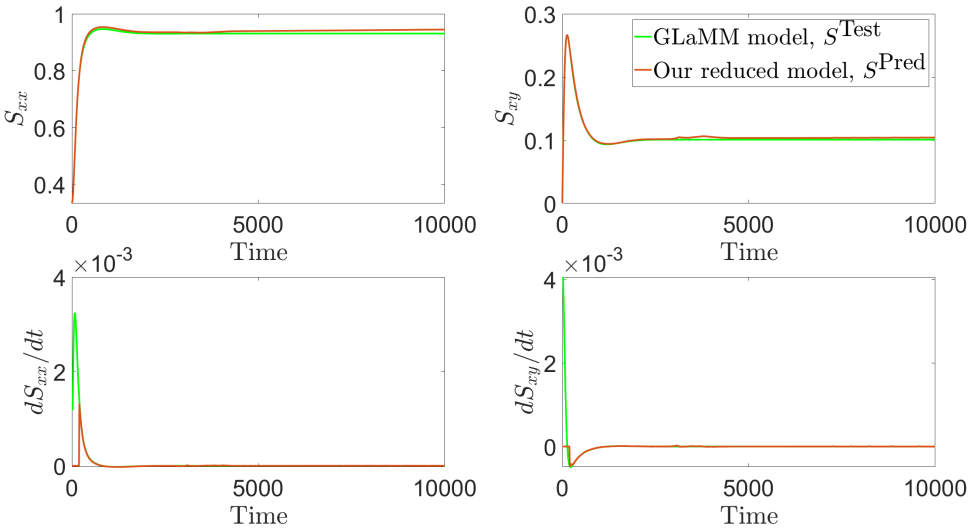


Figure 4.14: **S** tensor predictions from the new reduced model compared to the full GLaMM model solutions, for a shear flow of rate $\dot{\gamma}\tau_R = 9$. h_3 is the revised choice from equation (4.10) with $\alpha = 1$ and $\beta = 3$.

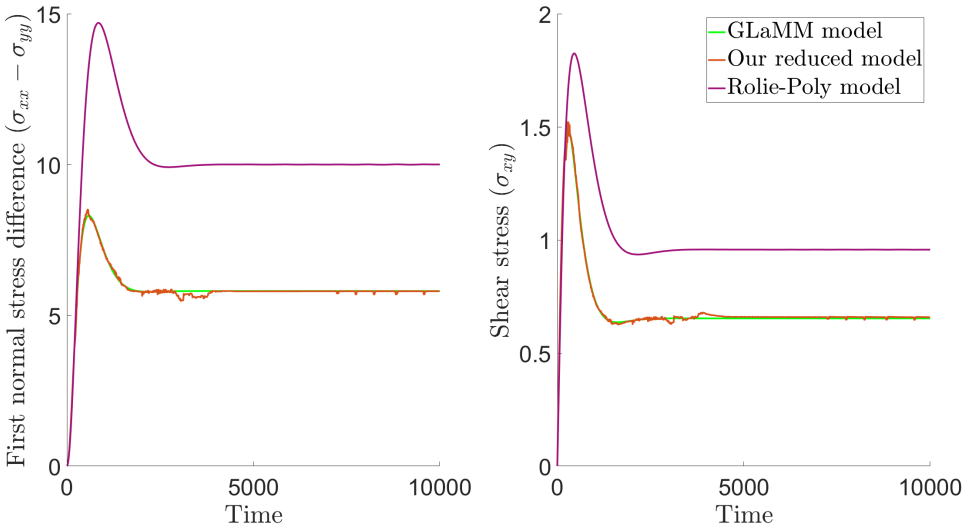


Figure 4.15: Stress predictions from the new reduced model compared to the full GLaMM model solutions and the Rolie-Poly model solutions, for a shear flow of rate $\dot{\gamma}\tau_R = 9$. The timestep used in the new reduced model is 900 times larger than the GLaMM model with a start time of $t = 200$. h_3 is the revised choice from equation (4.10) with $\alpha = 1$ and $\beta = 3$.

4.6.4 Remaining noise after revised Γ_3

We now have a model that is fundamentally sufficiently accurate, but the noise is not at an acceptable level. We suspect that the source of the noise is the derivatives of the functionals Γ_i. We isolate this by using our model to calculate the flow evolution, whilst replacing the predicted $\frac{\mathrm{d}}{\mathrm{d}t}\Gamma_i^{\mathrm{Pred}}$ with the true $\frac{\mathrm{d}}{\mathrm{d}t}\Gamma_i^{\mathrm{Test}}$ at every timestep for each functional in turn. Predicting with fixed $\frac{\mathrm{d}\Gamma_1}{\mathrm{d}t}$ or $\frac{\mathrm{d}\Gamma_2}{\mathrm{d}t}$ shows some improvement in the prediction of Γ_3, but no significant noise reduction. However, fixing $\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t}$ results in the plot shown in figure 4.16.

Figure 4.16 shows a drastic noise reduction in the prediction of $\frac{\mathrm{d}\Gamma_1}{\mathrm{d}t}$ and $\frac{\mathrm{d}\Gamma_2}{\mathrm{d}t}$ when data for $\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t}$ is directly provided to the model rather than calculated during the model evolution. The fact that such a significant improvement can be seen when only $\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t}$ is corrected implies that the sole cause of this noise issue is $\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t}$, not timestep size nor interpolation of Γ_1 and Γ_2. This

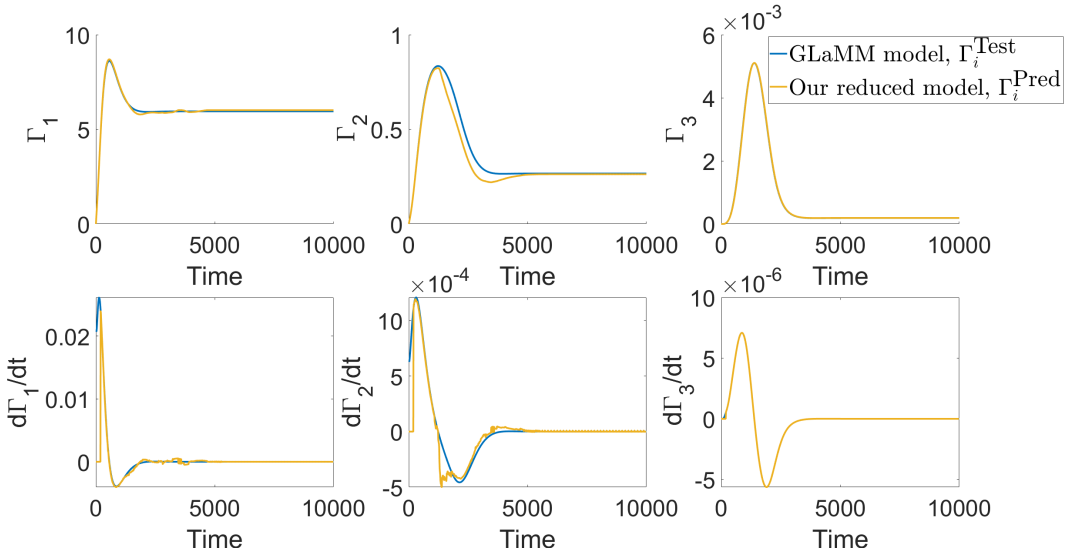


Figure 4.16: Γ_i predictions from the new reduced model compared to the full GLaMM model solutions, for a shear flow of rate $\dot{\gamma}\tau_R = 9$. h_3 is the revised choice from equation (4.10) with $\alpha = 1$ and $\beta = 3$. $\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t}$ is fixed to the result from GLaMM result data.

result also indicates that the choice of Γ_3 adequately characterises $\mathbf{f}(s, s')$, because given the correct $\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t}$, all three functionals evolve correctly. Thus we can conclude that the current Γ_3 from equation (4.10) is a suitable choice for our model, but its derivative $\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t}$ is sensitive to the changes in the predicted $\mathbf{f}^{\text{Pred}}_{\text{Prin}}$ configuration. Since a small change in location in $\mathbf{\Gamma}$ phase space has a relatively large effect on $\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t}$ it is difficult for the current interpolator to return a $\mathbf{f}^{\text{Pred}}_{\text{Prin}}$ that is accurate enough for the prediction of $\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t}$. These errors in $\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t}$ then propagate forward, which can errors in all three functionals during the next timestep. This is not a sufficient reason to change Γ_3 at this stage, provided that we address this sensitivity.

4.7 Adapting the interpolator for sparse data

4.7.1 Further noise reduction

One way that we could keep the same Γ_3 choice but improve the accuracy of $\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t}$ is to change the interpolation method we use to calculate $\mathbf{f}_{\mathrm{Prin}}^{\mathrm{Pred}}$. To do this we must first deduce how the octant interpolator causes noisy predictions. We can see an example of the octant interpolator's choice during steady state in figure 4.12. A potential source for $\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t}$ errors with the octant interpolator is the discrete nature by which it chooses to include a point or not. Let us consider the usage of our interpolator to predict two $\mathbf{f}_{\mathrm{Prin}}^{\mathrm{Pred}}$ profiles at similar but slightly different locations Γ_i in our phase space, such as is often the case with two subsequent timesteps. If the training data is sufficiently dense, there is a chance that our octant interpolator will choose a different set of eight points. If even one of these points is different, our interpolator considers the contribution from that entire region to have changed, potentially resulting in quite distinct $\mathbf{f}_{\mathrm{Prin}}^{\mathrm{Pred}}$ between the two locations. Whilst the effects on most of the functional quantities are minimal, the observed sensitivity of $\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t}$ can cause this value to differ, and in the case of subsequent timesteps, this manifests as rapid jumps in $\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t}$.

To confirm our hypothesis that all the noise in both $\frac{\mathrm{d}\Gamma_1}{\mathrm{d}t}$ and $\frac{\mathrm{d}\Gamma_2}{\mathrm{d}t}$ is induced by the noise in $\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t}$, let us analyse the noise during the steady state in figure 4.8, where all three functional derivatives suffer from noise. In this state, every component of $\frac{\partial}{\partial t}\mathbf{f}(s, s', t)$ is zero and thus each $\frac{\mathrm{d}\Gamma_i}{\mathrm{d}t}$ should also be zero. An error in $\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t}$ causes Γ_3 to be incorrect during the next timestep. As the model attempts to correct these errors so that $\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t} = 0$ it will create small errors in $\frac{\mathrm{d}\Gamma_1}{\mathrm{d}t}$ and $\frac{\mathrm{d}\Gamma_2}{\mathrm{d}t}$ due to the nature of the interpolator. This cycle repeats, and this self-correcting feedback will cause noise present in $\frac{\mathrm{d}\Gamma_3}{\mathrm{d}t}$ to

propagate into $\frac{d\Gamma_1}{dt}$ and $\frac{d\Gamma_2}{dt}$ and vice versa.

Both the rapid jumps in $\frac{d\Gamma_3}{dt}$ and the noise from the self-correcting feedback can be significantly reduced with an interpolator that employs a more continuous approach to interpolation. The current octant interpolator only uses eight points, one from each octant, and this can be quite sensitive with this sparser 3 functional training data as changing a single point that is used in the interpolation changes the contribution from that entire octant. This leads to the introduction of our "Radial octant average" interpolator which uses the advantages of both the radial and octant interpolators we have previously established. This interpolator evaluates the contribution from every point within a given radius r to reduce the dependence on any one point and exclude distant points that can dominate other contributions to the derivatives, but then creates one effective profile $(\bar{\mathbf{f}}^{\text{Train}}_{\text{Prin}})_q$ for each octant q to remove the interpolation bias that the radial interpolator often suffered from. Further details on the relevant formulas for this new interpolator can be found in appendix B.2.3. Calculations using this new interpolator should be much smoother, as rather than sharply switching the single point that represents an entire octant between successive timesteps, we average over multiple points to evaluate the contribution of an octant.

4.7.2 Shear prediction with the new interpolator

Using our new interpolator to repeat the predictions in figures 4.13-4.15, we evaluate the improvements when the new interpolator is used. The new prediction is plotted in figures 4.17-4.19.

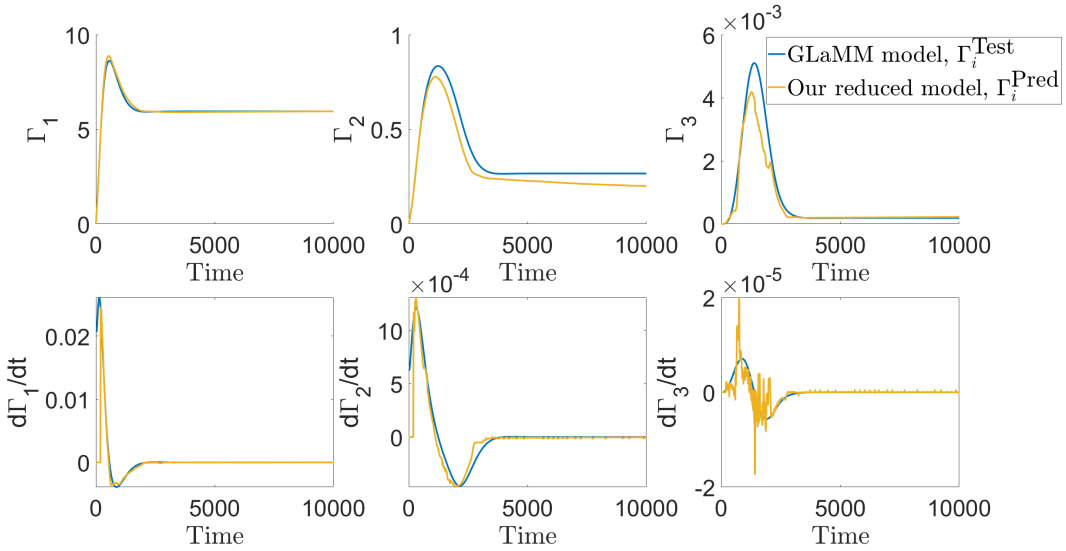


Figure 4.17: Γ_i predictions from the new reduced model with Runge-Kutta timestepping compared to the full GLaMM model solutions, for a shear flow of rate $\dot{\gamma}\tau_R = 9$. h_3 is the revised choice from equation (4.10) with $\alpha = 1$ and $\beta = 3$. The new radial octant average interpolator is used with $r = 1$.

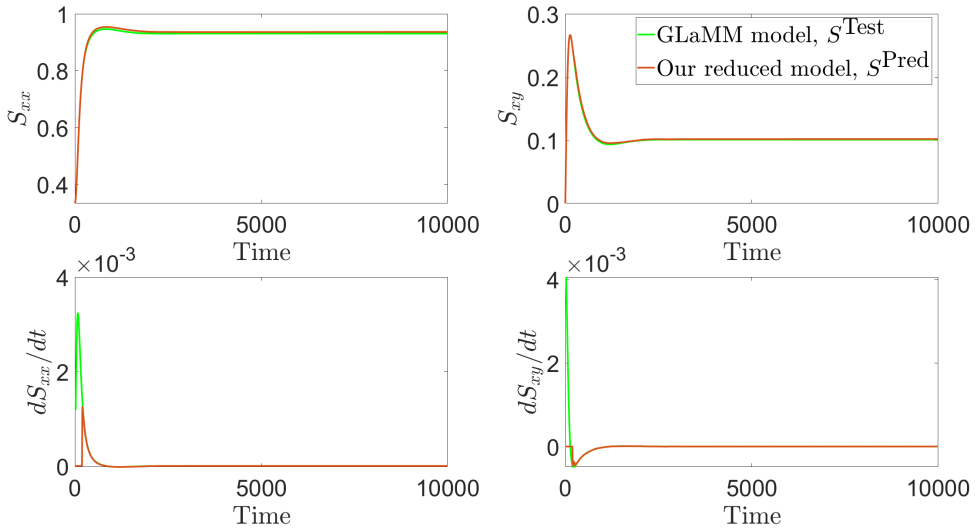


Figure 4.18: **S** tensor predictions from the new reduced model with Runge-Kutta timestepping compared to the full GLaMM model solutions, for a shear flow of rate $\dot{\gamma}\tau_R = 9$. h_3 is the revised choice from equation (4.10) with $\alpha = 1$ and $\beta = 3$. The new radial octant average interpolator is used with $r = 1$.

There is a clear improvement with the new interpolator, with a significant reduction in the noise of stress predictions in figure 4.19. This prediction used an interpolation radius of $r = 1$ and the Runge-Kutta method for timestep evolution. Whilst there is still some small discrepancy between

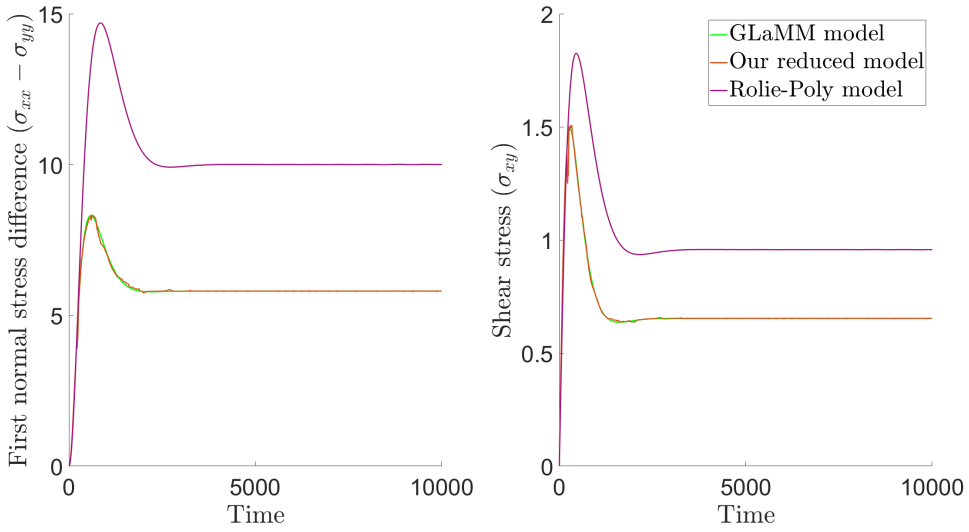


Figure 4.19: Stress predictions from the new reduced model with Runge-Kutta timestepping compared to the full GLaMM model solutions and the Rolie-Poly model solutions, for a shear flow of rate $\dot{\gamma}\tau_R = 9$. h_3 is the revised choice from equation (4.10) with $\alpha = 1$ and $\beta = 3$. The new radial octant average interpolator is used with $r = 1$.

the predictions and true values, and $\frac{\mathrm{d}}{\mathrm{d}t}\Gamma_3^{\mathrm{Pred}}$ has some noise, overall the functionals Γ_i are predicted sufficiently well that the stress is accurate. Further smoothing was considered, but an initial investigation showed no significant improvements and the model is now at an acceptable level of accuracy. This is our complete model, combining our results under extensional flows with considerations for rotation, a suitable third functional with corresponding extra dimension in the phase space, and the radial octant interpolator. With this, we now test our model under a range of different constant and non-constant shear rates.

4.8 Final model evaluation

With our final model established, we now evaluate its performance in predicting a range of constant and stepped shear rates alongside the Rolie-Poly model in figures 4.20 and 4.21.

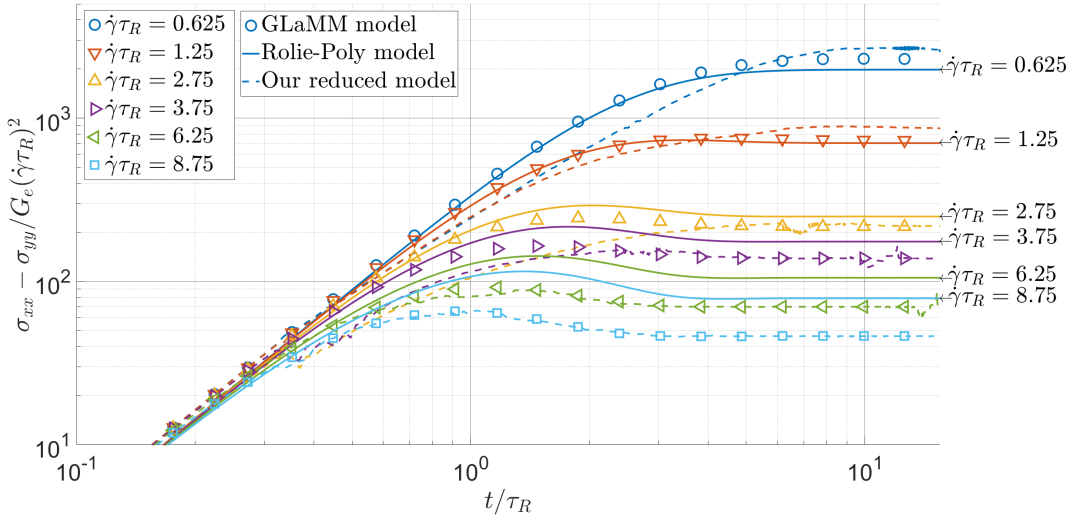


Figure 4.20: Stress predictions from the new reduced model with Runge-Kutta timestepping compared to the full GLaMM model solutions and the Rolie-Poly model solutions, for shear flows of various constant rates. h_3 is the revised choice from equation (4.10) with $\alpha = 1$ and $\beta = 3$. The new radial octant average interpolator is used with $r = 1$.

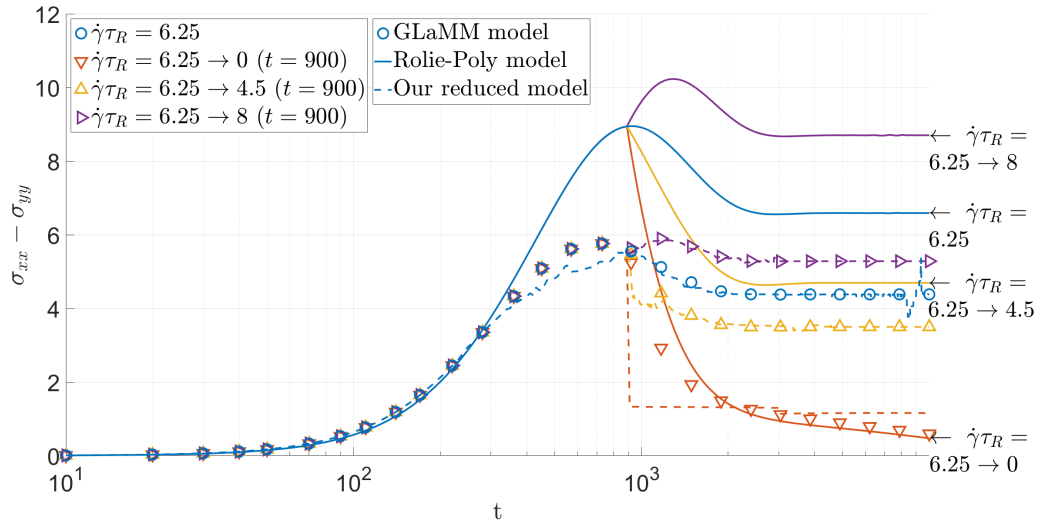


Figure 4.21: Stress predictions from the new reduced model with Runge-Kutta timestepping compared to the full GLaMM model solutions and the Rolie-Poly model solutions, for shear flows of various non-constant rates. h_3 is the revised choice from equation (4.10) with $\alpha = 1$ and $\beta = 3$. The new radial octant average interpolator is used with $r = 1$.

From figures 4.20 and 4.21 it is clear that our new reduced model performs well in a wide range of cases, outperforming the Rolie-Poly model by a significant margin in the majority of cases. The additional information from

our three-functional approach gives us a more accurate picture of the polymer dynamics than the single-mode Rolie-Poly model, whilst maintaining a significant advantage in speed over the GLaMM model due to our use of a larger timestep. Furthermore, our new reduced model calculates functional evolution directly with the GLaMM model, making it possible to predict data that this qualitatively different to training data such as the time-dependent shear in figure 4.21. This is significant because a purely data-driven approach would not be able to predict any flow that is of a different type from the ones it has been trained upon. One area where our reduced model does not perform as well as the Rolie-Poly model is shear of a low rate where the relative difference in functionals from one timestep to the next is much smaller, and our current choice of functionals is not able to sufficiently distinguish between these $\mathbf{f}(s, s')$ configurations. This is similar to the issues we highlighted in section 4.5 for small deformations. Our reduced model also struggles with a rate that drops to zero such as the one in figure 4.21 which is again a result of interpolation proving difficult at low strain.

4.9 Computational cost of the final model

Now that we have evaluated the accuracy of our final model, we now assess its computation speed when compared to the GLaMM model. Firstly we acknowledge that if you were to start from scratch, generating our training set and running our reduced model to predict a single flow, this would take significantly longer than evaluating the flow with the GLaMM model directly, as this would require you to simulate all 49 flows in our training set with the GLaMM model, compared to directly evaluating a single flow. However this is not the intended use for our reduced model, it is designed

in such a way that a new training set does not need to be generated each time. Once the training set has been generated a single time, it is saved and then can be used any number of times for different flows, with the high one-time cost of training set creation offset by the significant time saved by our reduced model over repeated calculations.

As has been shown multiple times throughout this thesis, such as in section 3.4.3, our reduced model is able to predict data that is qualitatively different to data in the training set. This indicates that our reduced model has an understanding of the underlying physics and means that the training set does not require more data to be added for each new feature. Our new model is also embeddable within finite element calculations for computational fluid dynamics, something that is not feasible for the GLaMM model due to its prohibitively high computation cost. This is also a perfect example of where the advantages of a single precomputed training set are unmistakable. The same training set we have already generated can be used in any number of finite element calculations, with each one evaluating our reduced model hundreds of times, at each point on the finite element mesh. An increased calculation speed also means that our reduced model is suitable for potential use with polydisperse polymers. A possible approach to bidisperse polymers is detailed in section 5.2.4.

To evaluate this speed improvement, we use both our reduced model and the GLaMM model to predict a single constant flow with shear rate of $\dot{\gamma}\tau_R = 9$, running both models on the same architecture with the same level of optimisation during compilation. Under these conditions, the GLaMM model took $35m7s$ on average whilst our reduced model completed calculation in an average time of $2m2s$, with repeated runs having variations of less than a second. Profiling the runtime of other flows displayed a similar level of reduction in calculation time. Of particular note is the fact that

if we break down the runtime of our reduced model further, the section of our code which calculates the $\frac{\partial \mathbf{f}}{\partial t}$ derivative using the GLaMM model accounted for $83\% - 85\%$ of our overall runtime, depending on the flow to be predicted. This indicates that all the additional calculations required for our reduced model such as training set interpolation do not significantly increase runtime. Our calculations in section 3.4.2 demonstrate that the GLaMM model requires more than 10000 times as many evaluations as our reduced model, implying a significant increase in speed if we were to further optimise our reduced model's code. Our model is currently written as a MATLAB function that interfaces with the existing GLaMM model, written in Fortran. Writing both models in the same programming language would likely improve our reduced model's calculation time and is a potential avenue to explore in future.

4.10 Comparison to experimental results

We also compare our reduced model results with experimental data [2]. To do this we evaluate the maximum shear stress σ_{xy}^{max} and maximum shear strain γ_{max} with our reduced model at various shear rates. Since these maximums are highly sensitive to noise, we fit a polynomial around the maximum to correctly identify the maximum for our reduced model. We compare these results to both experimental data and GLaMM model predictions in figure 4.22.

In figure 4.22 we can see that there is a bit of a disparity between the results from our reduced model and the GLaMM model, especially at lower rates. The errors in the lower rates are a consequence of the maxima being very weak at these rates, making them much less likely to remain intact in

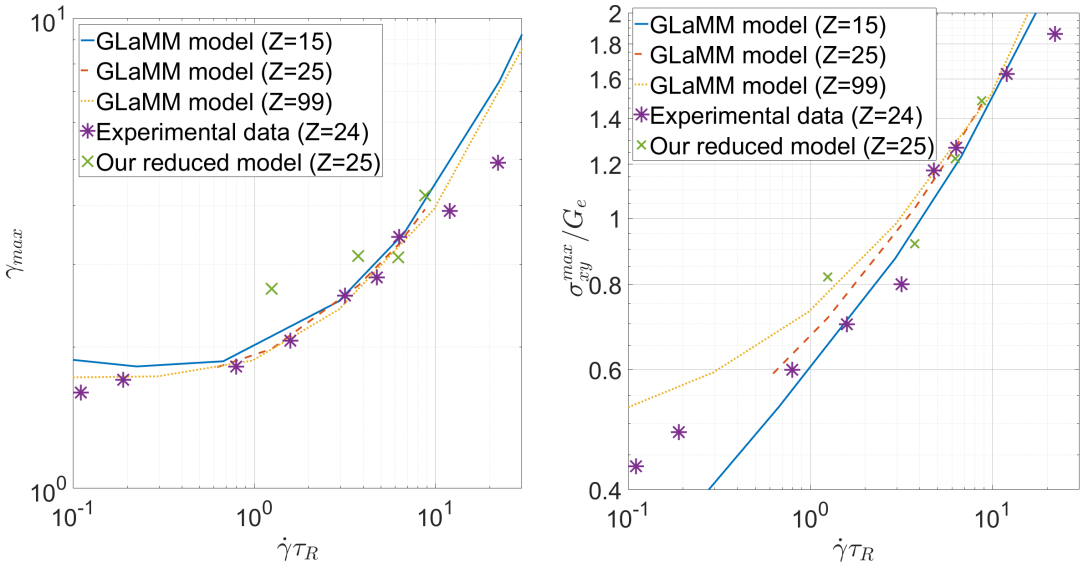


Figure 4.22: Maximum strain and maximum stress predictions comparing our reduced model and the GLaMM model with experimental data. The experimental data comes from a paper by Anwar and Graham [2], and these plots correspond to plots 8*b*) and 8*e*) from that paper.

our reduced model that predicts the GLaMM model. They also do not match the experimental results better than the GLaMM model, but this is to be expected as the aim of our reduced model, is not strictly to predict experimental data, it is to provide a better prediction of the GLaMM model than the Rolie-Poly model, which it achieves. However, as the construction of our reduced model framework permits the application to a wide range of PDEs, if an improvement is made to the GLaMM model that improves its agreement with experiments, or a new model is created that takes a similar form, the same steps outlined in this thesis can be applied to this new model, to produce a new reduced model that incorporates this new advancement.

4.11 Overview of the model in shear

In this chapter, we have made many enhancements to our reduced model from the extension flow chapter which generalised its use to be applicable to shear flows. We first introduce principal direction in order for our model to predict shear flows where polymer chains rotate as the flow progresses. With this, we then ensure that our training data has one uniform principal direction across all entries whenever we add data, and by tracking the evolution of principal direction during our predictions, this one data set can be used to predict flows of any principal direction. Subsequently, we quickly see that two functionals are insufficient to capture shear flows, and thus introduce a third functional. As we are able to capture uniaxial extension with two functionals, the new functional we introduce is the variation about principal direction, a quantity that is important to the dynamics of a polymer under shear, but has no relevance to a polymer under uniaxial extension flow. We also detail the natural generalisations that need to be made to the training set and interpolation methods when adding an extra dimension to the functional phase space. Results from predictions with three functionals are far more accurate, but the non-monotonicity and sensitivity of the variation about principal direction from equation (4.6) leads to significant numerical noise in predictions using this initial choice. To combat this noise, we change our choice for our third functional to a linear product combination to improve monotonicity and revise our interpolator for a smoother transition between timesteps as the radial octant average interpolator is able to slowly introduce and remove individual points much more effectively. All these advancements leave us with our final reduced model that is able to predict a wide range of both shear and uniaxial extension flows far more accurately than the Rolie-Poly model, at a much faster speed than the GLaMM model due to the larger timestep, all whilst

retaining the ability to predict flows with non-constant rates which are qualitatively different to the flows form the training set, a feature which is lost in purely data-driven approaches.

Chapter 5

Conclusions

5.1 Summary and discussion of results

The aim of this thesis was to produce a model with the speed of the Rolie-Poly model, but that achieves a similar level of accuracy as the GLaMM model. The GLaMM model has been shown to be close to experimental data, but there are a number of applications where it is not feasible to use the GLaMM model due to its computational cost and these areas would benefit greatly from a new, cheaper model. In this thesis we looked at shear and uniaxial extension flows as these are the major components required to perform computational fluid dynamics on a polymer. Some other areas that would benefit from a new model include polydispersity and flow-induced crystallisation [34]. For example, some research papers in polydispersity use the Rolie-Double-Poly which is based on, and thus inherits inaccuracies from, the Rolie-Poly model. Thus if the same process could be applied to the new reduced model from this thesis there would be a significant improvement in accuracy from the improved underlying model.

In order to create this model we first developed a model reduction framework for a specific class of PDEs. We developed a methodology for polymer dynamics, but one that is sufficiently general to be extendable to other PDEs with a time-dependent external driving term, such as the flow. The key assumption of this framework is there exists a collection of reduced variables $\mathbf{\Gamma}$ that can be calculated from the current model configuration $\mathbf{f}$ where any given value of $\mathbf{\Gamma}$ specifies cleanly the overall configuration $\mathbf{f}$. This is our coarse-graining assumption. Provided we are able to find a choice of $\mathbf{\Gamma}$ that is both slow-moving and obeys this assumption well for the chosen PDE, we can calculate the evolution of $\mathbf{\Gamma}$ more efficiently than evaluating $\mathbf{f}$ directly whilst being able to recover $\mathbf{f}$ from $\mathbf{\Gamma}$ at any time. Our model reduction method, like any direct data-driven method, uses a set of training data. However, unlike a direct data-driven method which generates a model solely on that data, our method uses the full configuration's PDE for the evolution of the dynamics, while the training set's role lies in the recovery of the full model configuration from the reduced variables. By determining the model's evolution directly from the PDE in this way, we have a level of physical intuition that would not be present if we relied on the training data alone. As a result of this, our method has the ability to predict behaviours outside of those upon which it has explicitly been trained, for example the prediction of variable rate flows from constant rate flows when applied to rheological PDEs.

With the general framework established, we applied this method to the GLaMM model for polymer dynamics in order to achieve our aim. The Rolie-Poly model is also a model reduction approach and reduces calculation complexity by only considering a single Rouse mode of the GLaMM model solution $\mathbf{f}(s, s', t)$. This does achieve its stated goal of a significant speed improvement over the GLaMM model, but this drastic over-

simplification results in a large amount of information about the model configuration $\mathbf{f}(s, s')$ being lost, restricting its ability to accurately predict polymer dynamics to a few of the simplest polymer flows. In contrast, our framework allows any choice of functionals to create a set of coarse-grained variables $\mathbf{\Gamma}$, giving much greater flexibility. When we created a new model for uniaxial extension, we also established the specifics of the training set and a suitable interpolator to predict a configuration $\mathbf{f}(s, s')$ at a given location in $\mathbf{\Gamma}$ phase space. The application to shear proceeded in much the same way, with the addition of a rotating frame of reference. This allowed our non-directional training data that has one uniform principal direction to predict flows that have any principal direction, including shear flows with non-constant principal direction. The code used for all of our reduced model predictions can be found on GitHub [28].

The results throughout this thesis show that our reduced model is significantly closer to the GLaMM model than the Rolie-Poly model and that our coarse-graining assumption can reproduce the stress to the required level of accuracy across a wide range of both uniaxial extension and shear flows. For constant rate flows, our reduced model produced accurate predictions for a wide range of uniaxial extension and shear flows, providing a significant improvement over the Rolie-Poly model for all flows that cause chain stretch. When we progressed to the prediction of time-dependent flows with non-constant rates, we observed a drastic improvement over the Rolie-Poly model for both uniaxial extension and shear flows. This ability to predict variable rate flows from a training set of constant rate flows is significant as it demonstrates that models with this framework are able to predict qualitatively different behaviour than that which they have been trained upon. This is something that is very difficult for a "black-box" model produced from a direct-data-driven approach and is made possible by our method's

utilisation of the GLaMM model PDE, which causes our model to retain a link to chain interactions on the molecular level. There are only two regions where our reduced model diverged from the GLaMM model predictions. In constant rate shear, our reduced model's accuracy only suffered at the lowest rates where chain stretch was minimal. This disparity was slightly more pronounced in shear flow prediction, with constant rate shear flows of Rouse Weissenberg number lower than 3 performing worse than the Rolie-Poly model. In variable rate flows, the prediction of dynamics during the final stages of relaxation back to the isotropic rest state, after the flow was removed, was difficult to capture correctly. The commonality that links these regions of inaccuracy is that the total stress is very low in both these regions. Problems that are confined to areas of very low total stress are not of particular concern as the Rolie-Poly model is sufficiently accurate when total stress is low, and if accuracy in this region is required, a system for transitioning between the Rolie-Poly model and our new reduced model is possible. Further exploration of this transition is left for future work.

In addition to our reduced model being accurate enough to satisfy this accuracy requirement, the computational cost of our reduced model is also a significant improvement over the GLaMM model. Despite our reduced model requiring the evaluation of the GLaMM model PDE at every timestep, much like the GLaMM model itself, our reduced model was able to maintain comparable accuracy with a timestep of 900 times the size of the GLaMM model timestep. Furthermore, the GLaMM model requires the PDE to be evaluated many times at each timestep, at each point on a mesh grid that spans s and s'. 1406 evaluations are required for a mesh size of $N = 75$. However with our framework, many functional choices, including those used in this thesis, do not require nearly as many PDE evaluations

as they only depend on a subset of the $\mathbf{f}(s, s')$ configuration. Overall, accounting for the extra cost incurred due to interpolation, our approach is still over 10000 times cheaper than the GLaMM model in computational cost.

With these results, we can confidently state that our new reduced model satisfies our aim of producing a new model for polymer dynamics that exists between the existing GLaMM and Rolie-Poly models, with speed comparable to the Rolie-Poly model, but accuracy far closer to the GLaMM model. Our results have shown that this improvement is the case across a range of uniaxial extension and shear flows, even for time-dependent flows that are qualitatively different to anything that is present in the training set. Due to the generality of the formulation, our framework is ready to test on a wide range of driven PDEs. The creation of other such models for other PDEs can follow the same approach we did in this thesis, the only change being that a new choice of reduced variables $\mathbf{\Gamma}$ will be required that uphold our coarse-graining assumption for this new PDE.

This brings us to the question of what makes a certain choice of functionals for $\mathbf{\Gamma}$ good. If we refer back to our coarse-graining assumption, it states that a suitable choice must produce a set of reduced variables $\mathbf{\Gamma}$ that univocally define $\mathbf{f}(\mathbf{s})$, such that for a specific $\mathbf{\Gamma}$, the $\mathbf{f}(\mathbf{s})$ configurations must be similar regardless of flow history. In addition, for a functional to be effective in this form of model reduction, the resulting $\mathbf{\Gamma}$ must be considerably slower moving than the configuration $\mathbf{f}(\mathbf{s})$ itself. Whilst this feature is not required by our coarse-graining assumption directly, if a reduced variable changes rapidly over time, it would not be possible to use a noticeably increased timestep compared to the direct computation of the PDE, and thus would not provide an appreciable improvement in computation speed. When we chose a set of functionals that possessed these two traits in this thesis, we

used the physical understanding of which quantities are important to define a polymer's behaviour at the macromolecular scale to produce suitable options, arriving at our final selection through a mixture of visual analysis of the resulting graphs and some trial and error. Whilst we cannot be sure that the functional choice we presented here is the best choice outside of the limited options we explored, it is a sufficiently effective choice to meet our challenging aim. Whilst for some other PDEs it may be possible to follow a similarly physically motivated route to arrive at an acceptable choice, a more general approach would be to use machine learning. Machine learning would allow us to evaluate a wider range of options at once, whilst also more rigorously determining the best choice available and further exploration is left as future work.

5.2 Future work

This thesis opens up several avenues for future research, some of which we have already begun to discuss. Areas we have identified for possible future investigation broadly fall into one of three types. One route would be to explore minor improvements to our reduced model for the GLaMM application such as noise reduction and functional optimisation. A second option would be to explore further uses of this reduced model for polymer dynamics such as polydisperse materials, flow-induced crystallisation and complex flows. Alternately, if the application of our reduced framework to models other than the GLaMM model is of interest, future work could provide improvements to our general model reduction framework such as using machine learning to automate functional choice. Each of these ideas and their significance shall be explained in this section.

5.2.1 Potential improvements to our reduced model

Firstly we shall address some of the potential minor improvements that could be made if they are necessary for future applications. One minor issue with our reduced model was noisy predictions. We reduced this noise to a level we could accept through the approaches in sections 4.6 and 4.7. However, certain applications may require a lower level of noise and for these cases, there is the potential for further smoothing. If a greater reduction to the noise is necessary, many approaches to average over multiple timesteps are available, each providing a different level of smoothing. One such method we considered was to average Γ_3 over multiple timesteps for the inverse transform only. This would reduce the overall noise as we have shown that the noise originates in Γ_3 and propagates to the other variables, without significantly affecting computation time. Another difficulty our reduced model encounters is a decrease in accuracy at low chain stretch. This occurs in both low-rate flows of constant rate and variable rate flows where the polymer is subjected to some type of flow, which is removed at a later time and the fluid is allowed to relax back towards the isotropic rest state. It is less important for our reduced model to capture these low-rate flows, as the Rolie-Poly model is already suitable for prediction of low chain stretch. In situations where a particular flow has periods of both high chain stretch and low chain stretch and both are required to a high degree of accuracy, we have proposed a cross-over between our reduced model and the Rolie-Poly model. To do so, we would first establish some criterion on the stress tensor, which would determine which model was more suitable at the current time. To transition between the two models we would also require a forward and backward mapping between the models. A mapping from our reduced model to the Rolie-Poly model is simple as we already have access to the stress tensor if $\mathbf{\Gamma}$ is known. For the other mapping, direct

calculation of $\boldsymbol{\Gamma}$ from $\boldsymbol{\sigma}$ is not possible, but we expect a simple empirical mapping will be sufficiently accurate at low $\boldsymbol{\sigma}$. With these mappings established, the resulting reduced model will be able to predict both low and high chain stretch dynamics well. Another aspect we have not yet explored is the prediction of polymer melts with a different chain length, Z. From our current results, we are optimistic that this should be possible immediately without any changes to our reduced model, though time constraints have not yet allowed us to confirm this. There are two routes we could take to achieve this. The natural way to achieve this for a number of discrete Z values would be to repeat the model reduction process multiple times, each time using GLaMM training data for a different value of Z. This way we would have a separate training set for each discrete value of Z, but very few changes would need to be made to the computation. Alternately, we could use the fact the GLaMM model's results are self-similar at high Weissenberg number to produce one training set that is usable for all Z. This would require more work, and the accuracy may suffer at lower Weissenberg number, but could lead to a more elegant solution for highly deformed chains where our reduced model already performs well. Accurate prediction of polymer chains of different lengths is important, and this is a requirement for the modelling of polydisperse polymers.

5.2.2 Possible future uses for our reduced model

Our model may be able to provide significant improvements to the prediction of polydisperse polymers. In this thesis, all predictions have been calculated for a monodisperse polymer melt, a material where all polymer chains are assumed to have the same length. In contrast, polydisperse materials are made up of a range of molecular weights with a molecular weight

distribution detailing the relative quantities. Monodisperse polymers are very rarely used for industrial production, virtually all industrial polymers are polydisperse. A way to calculate the dynamics for a polydisperse polymer is to discretise the molecular weight distribution into n distinct weights and calculate the interactions between each pair of polymer weights. This is the approach used in the Rolie-Double-Poly model [6], which details a framework for coupling together constitutive equations and handling constraint release. The Rolie-Double-Poly model uses the Rolie-Poly model as the base constitutive equations in this framework, and thus, the Rolie-Double-Poly model inherits the inaccuracies from the Rolie-Poly model. The GLaMM model was a possible choice for the base constitutive equations in this framework, but the framework requires the calculation of each pair of distinct weights individually, causing the calculation time to have scaling of n^2. With the computation time using the GLaMM model already substantial and the value of n often fairly large, this makes the GLaMM model a far less viable option for use in this framework. For this reason, the Rolie-Double-Poly paper chose to use the Rolie-Poly model over the GLaMM model despite its inaccuracies [6]. Our new reduced model would be suitable for application to this framework as the cost is comparable to the Rolie-Poly model, whilst inheriting many more features from the GLaMM model, significantly improving accuracy for high chain stretch. We explore a possible approach to applying our reduced model to bidisperse melts in section 5.2.4.

Another important application of models of polymer dynamics is flow-induced crystallisation (FiC) [34]. This crystallisation is intrinsically linked to a polymer's flow dynamics because flow enhances crystal nucleation. It is also essential for polymer processing as controlling the rate of crystal nucleation is required to produce the correct shape and properties of the

final product. Since virtually all processing and FiC experiments in the scientific literature [12] use polydisperse materials, modelling flow-induced crystallisation for these applications also requires the method to be appropriate for polydisperse polymers. For a polydisperse material, the longest chains deform the most and thus these chains contribute most strongly to nucleation [13]. The Rolie-Double-Poly model does not agree with the GLaMM model for highly deformed chains, and so accurately modelling flow-induced crystallisation requires an improved polydisperse model. Currently, the Rolie-Double-Poly is the most accurate polydisperse model, but we could utilise this framework with our reduced model as an improvement for the constitutive equations as described above, increasing applicability to highly deformed chains.

In this thesis, we have demonstrated the ability of our reduced model to predict uniaxial extension flows and shear flows in isolation, and so another natural direction for future work to proceed is the prediction of flows with both uniaxial extension and shear. Our predictions of variable rate flows have proven the ability of our reduced model to predict flows that are qualitatively different to those in the training set and so it should be possible to predict flows that have both uniaxial extension and shear components. Prediction of other complex flow types could also be explored [5]. Shear and uniaxial extension flows commonly appear in polymer processing, for example in a simple channel, shear often occurs at the edges of the channel as the polymer flow interacts with the walls, and uniaxial extension would occur if the channel were to narrow, increasing the speed of the flow. However, an industrial polymer process will rarely ever have one of these flow types in isolation, requiring a model that can predict both to model these flows. Complex flows usually use computational fluid dynamics which require the dynamics to be evaluated at all mesh points within

the flow field [39, 32]. The number of evaluations required means that the GLaMM model is not viable for such a computation, but the Rolie-Poly model is routinely used, which is of similar cost to our model. Combining a model that can predict flows with shear and uniaxial extension with the effects of polydispersity and crystallisation is required for the full picture.

5.2.3 Our framework with other driven PDEs and machine learning

A significant portion of this thesis has focused on the analysis of one specific reduced model for polymer dynamics when our model reduction framework is applied to the GLaMM model, but this framework could have many uses outside of polymer dynamics. Our framework is applicable to general externally driven PDEs. Other examples that such a framework could be applied to include constitutive equations for other materials, driven nonlinear diffusion and the nonlinear wave equation with an external driving term. The methodology of the framework should be directly applicable to these other PDEs, but there are a few possible challenges when choosing suitable functionals. For the GLaMM model, we chose a subset of functionals that were physically motivated based on their usefulness in describing polymer dynamics. The idea was that if these functionals could describe macroscale dynamics well, then two scenarios that have the same values for these functionals, and thus similar macroscale dynamics, are more likely to have similar molecular dynamics as well. Whilst this approach might be applicable to find good characterising functionals for some other PDEs, it will not be applicable to every case, and which physically motivated functionals are the correct ones to use may be very unclear for certain PDEs. A possible approach to make the choice of functionals more rigorous than

intuition alone is using machine learning to determine functionals.

Machine learning for functional choice could be performed to varying different degrees. If the PDE has sufficient physical interpretations that certain functionals of the PDE's dependent variable are established as useful quantities for the overall dynamics of the system, the user could propose a subset of potential functional choices, and the parameters for the best combination of these functionals are then learned by optimising an error metric. Alternatively, we could create a machine learning technique that searches for the functionals directly, inferring from the provided data and error metric. This second approach would require us to repose this as a machine learning problem, but the wider scope would allow us to find the ideal choice that an optimisation which was limited to parametrisation would have missed. Searching directly for the functional is also useful for other new PDEs where there is not a set of physical quantities as clear candidates. Both of these approaches would require an error metric, as a criterion to guide machine learning to good functional choices. The natural first option is to use the root mean square error between $\mathbf{f}_{\text{Lab}}^{\text{Pred}}$ and $\mathbf{f}_{\text{Lab}}^{\text{Test}}$, summed over s and s', as the error metric. Whilst this is a suitable option, certain regions of $\mathbf{f}$ are more important for an accurate prediction of polymer dynamics, and thus an error metric which weights these more strongly may make more suitable choices. We have previously seen the importance of accurate functional derivatives for our reduced model and using root mean square error in $\frac{\mathrm{d}\Gamma_i}{\mathrm{d}t}$ as a metric would ensure this feature is correct. A good metric that weights this region appropriately without neglecting overall accuracy could be some weighted sum between these two aforementioned metrics. Machine learning will give better answers more easily than intuition, so in addition to being a useful alternative when intuition is not an option, it can also be employed to improve upon reduced

models that choose functionals using intuition such as our reduced model. Whilst our physically motivated choice is sufficient to provide many accurate predictions for our GLaMM model reduction in this thesis, it has not been rigorously proven to be the best choice. Better functionals for our reduced model would provide further improvements to our model that was already sufficiently accurate to meet our aim.

5.2.4 Possible approach on how to use our model with bidisperse melts

In this subsection, we shall outline possible steps for the application of our model to polydisperse blends. Specifically, this section will look at bidisperse melts, melts that are a mixture of two distinct polymers with different lengths, Z.

As this thesis has focused on results for a monodisperse melt consisting solely of polymers with 25 entanglement segments, $Z = 25$, the first step to applying this to bidisperse melts would be to investigate how our model performs when predicting polymers with different values of Z. The quantities $\dot{\epsilon}\tau_R$ and $\dot{\gamma}\tau_R$, where $\tau_R = Z^2\tau_e$, are often utilised over $\dot{\epsilon}$ and $\dot{\gamma}$ because a polymer's response to a flow is highly dependant on the number of entanglement segments and the rouse time of a segment. This self-similarity between polymer flows with the same $\dot{\epsilon}\tau_R$ or $\dot{\gamma}\tau_R$ value is stronger at higher rates, which shows promise that our model may be able to predict polymers with different values of Z as it stands. If this is true, we can use our existing training set to predict flows with different values of Z. If not, we will simply create two training sets, one for each of the values of Z in the bidisperse melt.

The other change that we need to make for our reduced model to be applicable to a bidisperse melt is to reformulate the underlying GLaMM model so that it considers interactions between the two different lengths of polymers. We do this in a similar way to the formulation of the Rolie-Double-Poly model [6], denoting the quantities related to the shorter and longer of the two distinct chain lengths with subscripts "S" and "L", respectively. For a blend that contains a volume fraction ϕ_S of the shorter chains and ϕ_L of the longer chains, we have that,

$$\begin{aligned} \boldsymbol{\sigma}(t) &= \frac{12G_e}{5Z}\int_0^Z \mathbf{f}(s,s,t)\ ds \\ &= \frac{12G_e}{5Z}\left(\phi_S\int_0^Z \mathbf{f}_S(s,s,t)\ ds + \phi_L\int_0^Z \mathbf{f}_L(s,s,t)\ ds\right), \end{aligned} \quad (5.1)$$

and,

$$\mathbf{f}(s,s',t) = \phi_S\mathbf{f}_S(s,s',t) + \phi_L\mathbf{f}_L(s,s',t). \quad (5.2)$$

These equations mirror equations for the formulation of the Rolie-Double-Poly model with $\beta_{th} = 0$ and finite extensibility, $f_E(\lambda)$, assumed to be 1. We set the thermal constraint release parameter to 0 as this thermal contribution to CCR is already included in the ν term for the GLaMM model. The GLaMM model also assumes infinite extensibility, and choosing the finite extensibility function, $f_E(\lambda)$, to be 1 for all λ removes all effects of finite extensibility.

In equation (5.2), we have split $\mathbf{f}(s,s',t)$ into two distinct parts, the contribution from the short chains and the contribution from the long chains. As the dynamics of $\mathbf{f}(s,s',t)$ do not just depend on the length of the polymer chain itself, but also the length of the polymers it is entangled with, we further split the contributions, so that,

$$\mathbf{f}_S = \phi_S\mathbf{f}_{SS} + \phi_L\mathbf{f}_{SL}, \quad \mathbf{f}_L = \phi_S\mathbf{f}_{LS} + \phi_L\mathbf{f}_{LL}. \quad (5.3)$$

In this interpretation, each tensor field $\mathbf{f}_{IJ}$ accounts for the effects of entanglements of type J on the average configuration for chains of type I.

This leads to four equations, one for each $\mathbf{f}_{IJ}$, which are coupled using the same physics as the Rolie-Double-Poly model. Unlike the Rolie-Double-Poly model however, where the coupling is entirely controlled by the stretch ratio, λ, coupling within this new form of the GLaMM model appears in three distinct terms. The three terms that control the coupling for the GLaMM model are the CCR rate, ν, the effective number of entanglements under stretch, Z^*, and a new parameter we introduce, Θ, which is proportional to the stretch ratio.

We now define the evolution of each of these four components under the same form of the GLaMM model we have used in equation (2.3), by applying equations (5.2) and (5.3) to the relevant equations for the GLaMM model [14],

$$\begin{aligned}\frac{\partial \mathbf{f}_{SS}}{\partial t} &= \boldsymbol{\kappa}\cdot\mathbf{f}_{SS}+\mathbf{f}_{SS}\cdot\boldsymbol{\kappa}^T+\frac{1}{3Z_S\pi^2\tau_e}\left(\frac{Z_S}{Z_S^*(t)}\right)^2\left(\frac{\partial}{\partial s}+\frac{\partial}{\partial s'}\right)^2\mathbf{f}_{SS}\\ &\quad+\frac{3a\nu_S(t)}{2}\left[\frac{\partial}{\partial s}\left(\frac{a}{\Theta_S(s,t)}\frac{\partial}{\partial s}(\mathbf{f}_{SS}-\mathbf{f}^{eq})\right)\right.\\ &\quad\left.+\frac{\partial}{\partial s'}\left(\frac{a}{\Theta_S(s',t)}\frac{\partial}{\partial s'}(\mathbf{f}_{SS}-\mathbf{f}^{eq})\right)\right] \qquad (5.4)\\ &\quad+\frac{R_s}{2\pi^2\tau_e}\left[\frac{\partial}{\partial s}\left(\mathbf{f}_{SS}\frac{\partial}{\partial s}\ln[\Theta_S^2(s,t)]\right)+\frac{\partial}{\partial s'}\left(\mathbf{f}_{SS}\frac{\partial}{\partial s'}\ln[\Theta_S^2(s',t)]\right)\right],\end{aligned}$$

$$\begin{aligned}\frac{\partial \mathbf{f}_{SL}}{\partial t} &= \boldsymbol{\kappa}\cdot\mathbf{f}_{SL}+\mathbf{f}_{SL}\cdot\boldsymbol{\kappa}^T+\frac{1}{3Z_S\pi^2\tau_e}\left(\frac{Z_S}{Z_S^*(t)}\right)^2\left(\frac{\partial}{\partial s}+\frac{\partial}{\partial s'}\right)^2\mathbf{f}_{SL}\\ &\quad+\frac{3a\nu_L(t)}{2}\left[\frac{\partial}{\partial s}\left(\frac{a}{\Theta_S(s,t)}\frac{\partial}{\partial s}(\mathbf{f}_{SL}-\mathbf{f}^{eq})\right)\right.\\ &\quad\left.+\frac{\partial}{\partial s'}\left(\frac{a}{\Theta_S(s',t)}\frac{\partial}{\partial s'}(\mathbf{f}_{SL}-\mathbf{f}^{eq})\right)\right] \qquad (5.5)\\ &\quad+\frac{R_s}{2\pi^2\tau_e}\left[\frac{\partial}{\partial s}\left(\mathbf{f}_{SL}\frac{\partial}{\partial s}\ln[\Theta_S^2(s,t)]\right)+\frac{\partial}{\partial s'}\left(\mathbf{f}_{SL}\frac{\partial}{\partial s'}\ln[\Theta_S^2(s',t)]\right)\right],\end{aligned}$$

$$\begin{aligned}\frac{\partial \mathbf{f}_{LS}}{\partial t} &= \boldsymbol{\kappa}\cdot\mathbf{f}_{LS}+\mathbf{f}_{LS}\cdot\boldsymbol{\kappa}^T+\frac{1}{3Z_L\pi^2\tau_e}\left(\frac{Z_L}{Z_L^*(t)}\right)^2\left(\frac{\partial}{\partial s}+\frac{\partial}{\partial s'}\right)^2\mathbf{f}_{LS}\\ &+\frac{3a\nu_S(t)}{2}\left[\frac{\partial}{\partial s}\left(\frac{a}{\Theta_L(s,t)}\frac{\partial}{\partial s}(\mathbf{f}_{LS}-\mathbf{f}^{eq})\right)\right.\\ &\left.+\frac{\partial}{\partial s'}\left(\frac{a}{\Theta_L(s',t)}\frac{\partial}{\partial s'}(\mathbf{f}_{LS}-\mathbf{f}^{eq})\right)\right] \qquad (5.6)\\ &+\frac{R_s}{2\pi^2\tau_e}\left[\frac{\partial}{\partial s}\left(\mathbf{f}_{LS}\frac{\partial}{\partial s}\ln[\Theta_L^2(s,t)]\right)+\frac{\partial}{\partial s'}\left(\mathbf{f}_{LS}\frac{\partial}{\partial s'}\ln[\Theta_L^2(s',t)]\right)\right],\end{aligned}$$

$$\begin{aligned}\frac{\partial \mathbf{f}_{LL}}{\partial t} &= \boldsymbol{\kappa}\cdot\mathbf{f}_{LL}+\mathbf{f}_{LL}\cdot\boldsymbol{\kappa}^T+\frac{1}{3Z_L\pi^2\tau_e}\left(\frac{Z_L}{Z_L^*(t)}\right)^2\left(\frac{\partial}{\partial s}+\frac{\partial}{\partial s'}\right)^2\mathbf{f}_{LL}\\ &+\frac{3a\nu_L(t)}{2}\left[\frac{\partial}{\partial s}\left(\frac{a}{\Theta_L(s,t)}\frac{\partial}{\partial s}(\mathbf{f}_{LL}-\mathbf{f}^{eq})\right)\right.\\ &\left.+\frac{\partial}{\partial s'}\left(\frac{a}{\Theta_L(s',t)}\frac{\partial}{\partial s'}(\mathbf{f}_{LL}-\mathbf{f}^{eq})\right)\right] \qquad (5.7)\\ &+\frac{R_s}{2\pi^2\tau_e}\left[\frac{\partial}{\partial s}\left(\mathbf{f}_{LL}\frac{\partial}{\partial s}\ln[\Theta_L^2(s,t)]\right)+\frac{\partial}{\partial s'}\left(\mathbf{f}_{LL}\frac{\partial}{\partial s'}\ln[\Theta_L^2(s',t)]\right)\right],\end{aligned}$$

where,

$$\Theta_S(s,t)=\sqrt{\operatorname{Tr}\mathbf{f}_S(s,s,t)},\quad \Theta_L(s,t)=\sqrt{\operatorname{Tr}\mathbf{f}_L(s,s,t)}, \qquad (5.8)$$

$$Z_S^*(t)=\frac{1}{a}\int_0^Z\Theta_S(s,t)\,ds,\quad Z_L^*(t)=\frac{1}{a}\int_0^Z\Theta_L(s,t)\,ds, \qquad (5.9)$$

$$\nu_S(t)=c_\nu\left(\Lambda_S(t)+\frac{4}{Z_SZ_S^*(t)\pi^2\tau_e}\right),\quad \nu_L(t)=c_\nu\left(\Lambda_L(t)+\frac{4}{Z_LZ_L^*(t)\pi^2\tau_e}\right), \qquad (5.10)$$

and,

$$\begin{aligned}\Lambda_S(t) &= -\frac{1}{Z_S^*(t)}\frac{\partial}{\partial t}\int_0^Z\left.\frac{\sqrt{\operatorname{Tr}\mathbf{f}_S(s,s,t)}}{a}\right|_{Retraction}ds\\ &= -\frac{1}{2aZ_S^*(t)}\int_0^Z\frac{1}{\Theta_S(s,t)}\left(\operatorname{Tr}\left[\phi_S\frac{\partial\mathbf{f}_{SS}}{\partial t}+\phi_L\frac{\partial\mathbf{f}_{SL}}{\partial t}\right]\right)\Bigg|_{Retraction}ds\\ &= -\frac{R_s}{2aZ_S^*(t)\pi^2\tau_e}\frac{\partial}{\partial t}\int_0^Z\frac{1}{\Theta_S(s,t)}\frac{\partial}{\partial s}\Big(\operatorname{Tr}\left[\phi_S\mathbf{f}_{SS}(s,s',t)\right.\\ &\left.+\phi_L\mathbf{f}_{SL}(s,s',t)\right]\frac{\partial}{\partial s}\ln[\Theta_S^2(s,t)]\Big)\Big|_{s'=s}ds, \qquad (5.11)\end{aligned}$$

$$\begin{aligned}
\Lambda_L(t) &= -\frac{1}{Z_L^*(t)}\frac{\partial}{\partial t}\int_0^Z \frac{\sqrt{\mathrm{Tr}\,\mathbf{f}_L(s,s,t)}}{a}\bigg|_{Retraction} ds \\
&= -\frac{1}{2aZ_L^*(t)}\int_0^Z \frac{1}{\Theta_L(s,t)}\left(\mathrm{Tr}\left[\phi_S\frac{\partial \mathbf{f}_{LS}}{\partial t}+\phi_L\frac{\partial \mathbf{f}_{LL}}{\partial t}\right]\right)\bigg|_{Retraction} ds \\
&= -\frac{R_s}{2aZ_L^*(t)\pi^2\tau_e}\frac{\partial}{\partial t}\int_0^Z \frac{1}{\Theta_L(s,t)}\frac{\partial}{\partial s}\Bigg(\mathrm{Tr}\left[\phi_S\mathbf{f}_{LS}(s,s',t)\right. \\
&\quad \left.+\phi_L\mathbf{f}_{LL}(s,s',t)\right]\frac{\partial}{\partial s}\ln[\Theta_L^2(s,t)]\Bigg)\bigg|_{s'=s} ds. \qquad (5.12)
\end{aligned}$$

In this formulation, we have terms to represent the effects of convection, reptation, CCR and retraction, and the similarities between this formulation in equations (5.5)-(5.8) in this thesis and equations (16)-(19) in the Rolie-Double-Poly paper [6] are evident when you set $\beta_{th} = 0$ and $f_E(\lambda) = 1$. In this formulation of the GLaMM model for bidisperse melts, the type of chain to use for each of the coupled terms ν, Z^* and Θ is not immediately obvious and an explanation requires the physics of the different methods by which a polymer chain can relax. Both the reptation and retraction terms each only has one variable which is coupled to the other $\mathbf{f}_{IJ}$ equations, Z^* and Θ respectively. Both of these relaxation methods depend on the movement of the polymer chain within its constraining tube, and thus the coupled variables within the reptation and retraction terms will use chains of type I. However, CCR occurs when the polymer chains that form the surrounding tube itself rearrange, lessening the constraints on the polymer chain within the tube. $\mathbf{f}_{IJ}$ considers the effects of entanglements of type J on the average configuration of the type I chains, and so this CCR rate, ν, will depend on chains of type J. In addition, the effects of CCR on a polymer chain are diminished by how stretched the chain is, represented by $\frac{a}{\Theta}$ in this form of the GLaMM model, and so this coupling will use chains of type I. Taking the "short-long" chain interaction, $\mathbf{f}_{SL}$ as an example, reptation and retraction are calculated using the shorter chains of length Z_S, as the rates of reptation and retraction depend on the

length of the polymer chain, whilst the rate of convective constraint release depends upon the length of the chains that the constraining tube is comprised of, and the effect is then reduced in proportion to how stretched the shorter chains are. In this example of "short-long" chain interaction, the entangling chains are considered to be of the longer type, and so the CCR rate is calculated with this longer length Z_L. As a result, the coupling for $\frac{\partial}{\partial t}\mathbf{f}_{SL}$ in equation (5.6) is controlled by Z_S^*, Θ_S and ν_L.

Using these equations we can now apply our reduced model methodology to bidisperse melts. Each of the four components will have its own set of functionals $\mathbf{\Gamma}_{IJ}$, from which we can predict each of the four components, $\mathbf{f}_{IJ}$, from the training data, using equations (5.5)-(5.8) to calculate $\frac{\partial}{\partial t}\mathbf{f}_{IJ}$, and equations (4.3), (4.4) and (4.10) to calculate the set of $\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{\Gamma}_{IJ}$ for each component. How to predict $\mathbf{f}_{IJ}$ from the training data depends on whether our existing training set can reliably predict the evolution of polymer chains with a range of different Z values. If it can, this training set can be used for all four components. If not, two new training sets will need to be created, one for $Z = Z_S$ and one for $Z = Z_L$. The first index, I, of each $\mathbf{f}_{IJ}$ component indicates the polymer chain length of that component and thus determines which training set to use. And so, for this case where two training sets are required, $\mathbf{f}_{SS}$ and $\mathbf{f}_{SL}$ will be predicted with the training set where $Z = Z_S$, whilst $\mathbf{f}_{LS}$ and $\mathbf{f}_{LL}$ will be predicted with the second training set where $Z = Z_L$.

Bibliography

[1] J. M. Adams, S. M. Fielding, and P. D. Olmsted. Transient shear banding in entangled polymers: A study using the Rolie-Poly model. *Journal of Rheology*, 55(5):1007–1032, 09 2011.

[2] M. Anwar and R. S. Graham. Nonlinear shear of entangled polymers from nonequilibrium molecular dynamics. *Journal of Polymer Science Part B: Polymer Physics*, 57(24):1692–1704, 2019.

[3] L. Attfield. Polymer dynamics investigation. Unpublished master's dissertation, University of Nottingham, 2018.

[4] P. Benner, S. Gugercin, and K. Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM Review*, 57(4):483531, 2015.

[5] G. Bishko, T. C. B. McLeish, O. G. Harlen, and R. G. Larson. Theoretical molecular rheology of branched polymers in simple and complex flows: The pom-pom model. *Physical Review Letters*, 79:2352–2355, Sep 1997.

[6] V. A. Boudara, J. D. Peterson, L. Gary Leal, and D. J. Read. Nonlinear rheology of polydisperse blends of entangled linear polymers: Rolie-double-poly models. *Journal of Rheology*, 63(1):7191, 2019.

[7] D. Dabiri, M. Saadat, D. Mangal, and S. Jamali. Fractional rheology-

informed neural networks for data-driven identification of viscoelastic constitutive models. *Rheologica Acta*, 62(10):557–568, Oct 2023.

[8] M. Doi and S. F. Edwards. Dynamics of concentrated polymer systems. part 1.brownian motion in the equilibrium state. *J. Chem. Soc., Faraday Trans. 2*, 74:17891801, 1978.

[9] B. E. Dolata and P. D. Olmsted. A thermodynamically consistent constitutive equation describing polymer disentanglement under flow. *Journal of Rheology*, 67(1):269–292, 01 2023.

[10] S. Farrington, S. Jariwala, M. Armstrong, E. Nigro, N. J. Wagner, and A. N. Beris. Physiology-based parameterization of human blood steady shear rheology via machine learning: A hemostatistics contribution. *Rheologica Acta*, 62(10):491–506, Oct 2023.

[11] D. Givon, R. Kupferman, and A. Stuart. Extracting macroscopic dynamics: Model problems and algorithms. *Nonlinearity*, 17(6), 2004.

[12] R. S. Graham. Modelling flow-induced crystallisation in polymers. *Chemical Communications*, 50:3531–3545, 2014.

[13] R. S. Graham. Understanding flow-induced crystallization in polymers: A perspective on the role of molecular simulations. *Journal of Rheology*, 63(1):203–214, 01 2019.

[14] R. S. Graham, A. E. Likhtman, T. C. McLeish, and S. T. Milner. Microscopic theory of linear, entangled polymer chains under rapid deformation including chain stretch and convective constraint release. *Journal of Rheology*, 47(5):11711200, 2003.

[15] C. Hannecart, T. Shahid, D. Vlassopoulos, F. Oosterlinck, C. Clasen, and E. van Ruymbeke. Decoding the steady elongational viscosity of

monodisperse linear polymers using tube-based modeling. *Journal of Rheology*, 66(1):197–218, 01 2022.

[16] A. A. Howard, J. Dong, R. Patel, M. D'Elia, M. R. Maxey, and P. Stinis. Machine learning methods for particle stress development in suspension poiseuille flows. *Rheologica Acta*, 62(10):507–534, Oct 2023.

[17] G. Ianniruberto and G. Marrucci. A simple constitutive equation for entangled polymers with chain stretch. *Journal of Rheology*, 45(6):1305–1318, 11 2001.

[18] H. Jin, S. Yoon, F. C. Park, and K. H. Ahn. Data-driven constitutive model of complex fluids using recurrent neural networks. *Rheologica Acta*, 62(10):569–586, Oct 2023.

[19] S. Kang, H. Jin, C. H. Ahn, J. Nam, and K. H. Ahn. Classification of battery slurry by flow signal processing via echo state network model. *Rheologica Acta*, 62(10):605–615, Oct 2023.

[20] K. R. Lennon, J. D. J. Rathinaraj, M. A. Gonzalez Cadena, A. Santra, G. H. McKinley, and J. W. Swan. Anticipating gelation and vitrification with medium amplitude parallel superposition (maps) rheology and artificial neural networks. *Rheologica Acta*, 62(10):535–556, Oct 2023.

[21] A. E. Likhtman and R. S. Graham. Simple constitutive equation for linear polymer melts derived from molecular theory: Roliepoly equation. *Journal of Non-Newtonian Fluid Mechanics*, 114(1):112, 2003.

[22] A. E. Likhtman and T. C. McLeish. Quantitative theory for linear dynamics of linear entangled polymers. *Macromolecules*, 35(16):63326343, 2002.

[23] M. Mahmoudabadbozchelou, M. Caggioni, S. Shahsavari, W. H. Hartt, G. Em Karniadakis, and S. Jamali. Data-driven physics-informed constitutive metamodeling of complex fluids: A multifidelity neural network (mfnn) framework. *Journal of Rheology*, 65(2):179198, 2021.

[24] C. McIlroy and P. D. Olmsted. Disentanglement effects on welding behaviour of polymer melts during the fused-filament-fabrication method for additive manufacturing. *Polymer*, 123:376–391, 2017.

[25] D. W. Mead, R. G. Larson, and M. Doi. A molecular theory for fast flows of entangled polymers. *Macromolecules*, 31(22):7895–7914, 1998.

[26] D. W. Mead, S. Monjezi, and J. Park. A constitutive model for entangled polydisperse linear flexible polymers with entanglement dynamics and a configuration dependent friction coefficient. Part I: Model derivation. *Journal of Rheology*, 62(1):121–134, 01 2018.

[27] D. Mellor. Polymer dynamics investigation. Unpublished master's dissertation, University of Nottingham, 2019.

[28] D. Mellor. Reduced polymer model. https://github.com/8dan8/Reduced-polymer-model, 2024.

[29] N. Moreno, S. P. Nunes, and V. M. Calo. Consistent model reduction of polymer chains in solution in dissipative particle dynamics: Model description. *Computer Physics Communications*, 196:255266, 2015.

[30] E. Muravleva, I. Oseledets, and D. Koroteev. Application of machine learning to viscoplastic flow modeling. *Physics of Fluids*, 30(10):103102, 2018.

[31] M. H. Nafar Sefiddashti, B. J. Edwards, and B. Khomami. Elucidating the molecular rheology of entangled polymeric fluids via compar-

ison of atomistic simulations and model predictions. *Macromolecules*, 52(21):8124–8143, 2019.

[32] R. G. Owens and T. N. Phillips. *Computational Rheology*. World Scientific Publishing Company, 2002.

[33] C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning*. MIT Press, 2006.

[34] D. J. Read, C. McIlroy, C. Das, O. G. Harlen, and R. S. Graham. Polystrand model of flow-induced nucleation in polymers. *Physical Review Letters*, 124:147802, Apr 2020.

[35] B. Robertson, R. L. Thompson, T. C. B. McLeish, and I. Robinson. Polymer extrudate-swell: From monodisperse melts to polydispersity and flow-induced reduction in monomer friction. *Journal of Rheology*, 63(2):319–333, 03 2019.

[36] P. S. Stephanou, I. Ch. Tsimouri, and V. G. Mavrantzas. Simple, accurate and user-friendly differential constitutive model for the rheology of entangled polymer melts and solutions from nonequilibrium thermodynamics. *Materials*, 13(12), 2020.

[37] H. Taghipour, S. Costanzo, D. Vlassopoulos, E. van Ruymbeke, and L. G. D. Hawke. Entangled linear polymers in fast shear flows: Comparison of tube-model predictions and experimental data. *Journal of Rheology*, 65(6):1111–1137, 11 2021.

[38] H. Weeratunge, D. Robe, A. Menzel, A. W. Phillips, M. Kirley, K. Smith-Miles, and E. Hajizadeh. Bayesian coarsening: rapid tuning of polymer model parameters. *Rheologica Acta*, 62(10):477–490, Oct 2023.

[39] H. G. Weller, G. Tabor, H. Jasak, and C. Fureby. A tensorial approach to computational continuum mechanics using object orientated techniques. *Computers in Physics*, 12:620–631, 11 1998.

[40] C. D. Young, P. T. Corona, A. Datta, M. E. Helgeson, and M. D. Graham. Scattering-informed microstructure prediction during lagrangian evolution (simple)—a data-driven framework for modeling complex fluids in flow. *Rheologica Acta*, 62(10):587–604, Oct 2023.

[41] L. Zhao, Z. Li, B. Caswell, J. Ouyang, and G. Em Karniadakis. Active learning of constitutive relation from mesoscopic dynamics for macroscopic modeling of non-newtonian flows. *Journal of Computational Physics*, 363:116127, 2018.

Appendices

Appendix A

Fourier Investigation

Prior to this research, we had explored the possibility of a "Rolie-Poly Deluxe" model [27] that would build upon the existing Rolie-Poly model with the same aim as this research; to produce an intermediate model that would fit between the existing GLaMM and Rolie-Poly models.

The GLaMM model can be split into contributions from N^2 distinct Fourier modes, and the Rolie-Poly model is a model reduction that predicts the largest mode only and ignores the rest. The more Fourier modes used in a numerical calculation the greater the accuracy but the slower the computation speed, leading to the disparity between the GLaMM and Rolie-Poly models. Our prior work was to try to establish a formula that predicts the size of the other modes from the size of the largest mode. This way, no other modes would need to be directly calculated, but predicting these modes would provide a significant accuracy improvement over neglecting them entirely.

In such an approach, we first take a 2 dimensional Fourier decomposition of the GLaMM model in s and s'. We then define the Fourier modes, $\mathbf{C}_{pq}(t)$,

using the following formula, equation (A.1).

$$\mathbf{C}_{pq}(t) = \int_0^Z \int_0^Z [\mathbf{f}(s,s',t) - \mathbf{f}_{eq}(s,s',t)] \sin\left(\frac{\pi p s}{Z}\right) \sin\left(\frac{\pi q s'}{Z}\right) ds ds'. \quad \text{(A.1)}$$

Under this formulation, $\mathbf{C}_{11}$ is a 3 by 3 tensor containing the $p = q = 1$ mode for each Cartesian coordinate pair. As with regular Fourier decomposition, we can produce an estimate of the function $\mathbf{f}(s,s',t) - \mathbf{f}_{eq}(s,s',t)$ from a few Fourier modes. We show a prediction of a typical $\mathbf{f}(s,s')$ profile at a snapshot in time under uniaxial extension flow using Fourier modes in figure A.1.

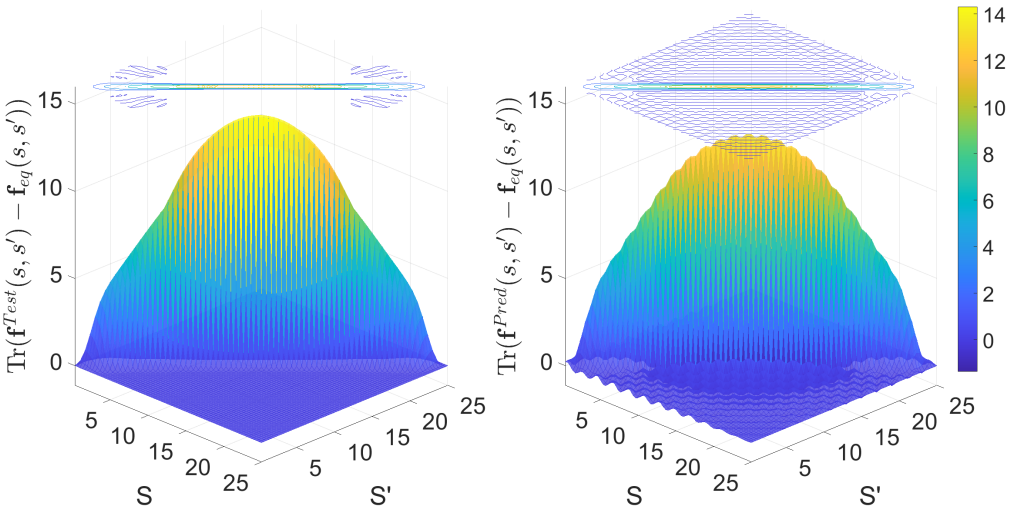


Figure A.1: Comparison between a true profile (left) and a prediction constructed from a few of its Fourier modes (right).

In figure A.1, we intentionally choose an insufficient number of terms to make it clear how the estimate approaches the true value as terms are added. The Rolie-Poly model was created around this idea, using the largest mode, $\mathbf{C}_{11}$, alone to predict polymer response, whilst assuming all other $\mathbf{C}_{pq}$ modes are zero. This neglection of the higher Fourier modes directly contributes to the significant loss of accuracy the Rolie-Poly model suffers under many polymer experiments, as shown in figure A.2.

Figure A.2 shows a clear discrepancy between the viscosity predictions, η,

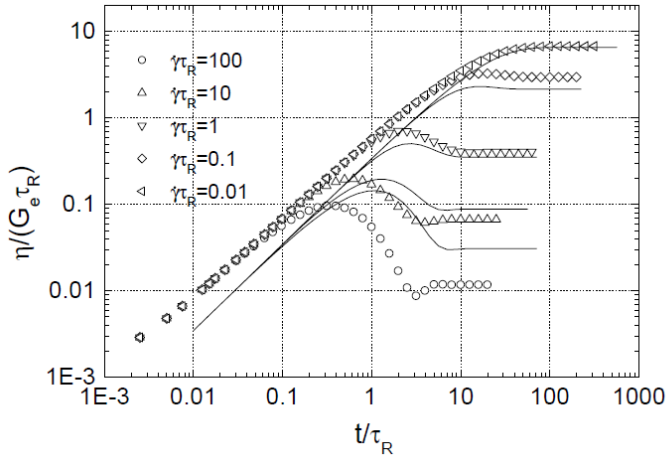


Figure A.2: Transient start-up shear predictions of the GLaMM model (symbols) and Rolie-Poly equation (lines) for $Z = 20$, from the Rolie-Poly paper [21].

for the two models. One natural extension when faced with this problem is successively to add more modes until an acceptable degree of accuracy is reached. However when this was attempted in practice, a very large number of modes was required for a suitable prediction, making the creation of a fast model that evaluates multiple modes unfeasible. Our proposed solution was to find a solution to $\mathbf{C}_{pq} \approx M(p,q)\mathbf{C}_{11}$ for some function $M(p,q)$ that was consistent across different flow rates for uniaxial extension. Such an $M(p,q)$ could predict the amplitudes of the other $\mathbf{C}_{pq}$ mode whilst only requiring direct calculation of $\mathbf{C}_{11}$, improving prediction accuracy with more modes whilst retaining the speed of the Rolie-Poly model.

To find Mp,q we investigated the relative size of each Fourier mode. Our prior work was in agreement with other research [3] that the Fourier modes on the diagonal, $\mathbf{C}_{pq}\ for\ (p = q)$, had the biggest influence on prediction accuracy, with $\mathbf{C}_{11}$ having the largest influence of all. However, a new discovery of note in our prior work was that an accurate prediction necessitated the consideration of all modes within a diagonal stripe such as $|p - q| \leq 4$. Our prior work attempted to create a $M(p,q)$ with a focus on greater accuracy when the difference between p and q was small, but no

suitable $M(p,q)$ was found.

Further investigation at the beginning of this research in this area led to the counter-example in figure A.3.

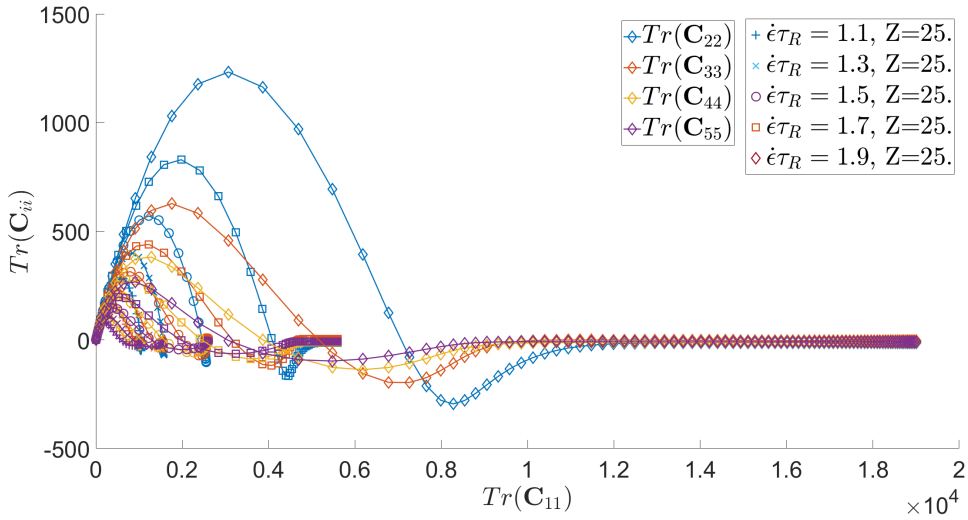


Figure A.3: A plot showing the relation between the trace of the amplitudes for the diagonal $\mathbf{C}_{ii}$ modes and the $\mathbf{C}_{11}$ mode for uniaxial extension at different rates.

Figure A.3 shows that the relation is not consistent between different uniaxial extension rates, and thus a suitable $M(p,q)$ does not exist. A relation of the form $M^*(p,q,\mathbf{C}_{11},...,\mathbf{C}_{ii})$ for some chosen integer $i > 1$ may have led to a definition that is consistent across different rates, but this would require the calculation of more than one Fourier mode. Whilst this may be possible, the quantity of Fourier modes that were often required for an accurate prediction implied that a Fourier decomposition of $\mathbf{f}(s,s',t)$ may not be the most efficient basis to reduce the GLaMM model. Thus, we decided not to restrict ourselves to only Fourier modes, opting for the more general structure presented in section 3.1.

Appendix B

Interpolation

B.1 2D Interpolation

In order to utilise our new method we need to able to produce $\mathbf{f}^{\text{Pred}}$ from knowledge of Γ_i^{Pred} using the training data. This leads us to consider the different options for interpolation in Γ_i space.

B.1.1 Quadrant interpolator

Firstly we introduce the method of our quadrant interpolator for $k = 2$. For a chosen $\Gamma_1^{\text{Pred}}, \Gamma_2^{\text{Pred}}$ we then choose the closest points in $\mathbf{\Gamma}$ space from the training set trajectories in each quadrant as shown in figure B.2.

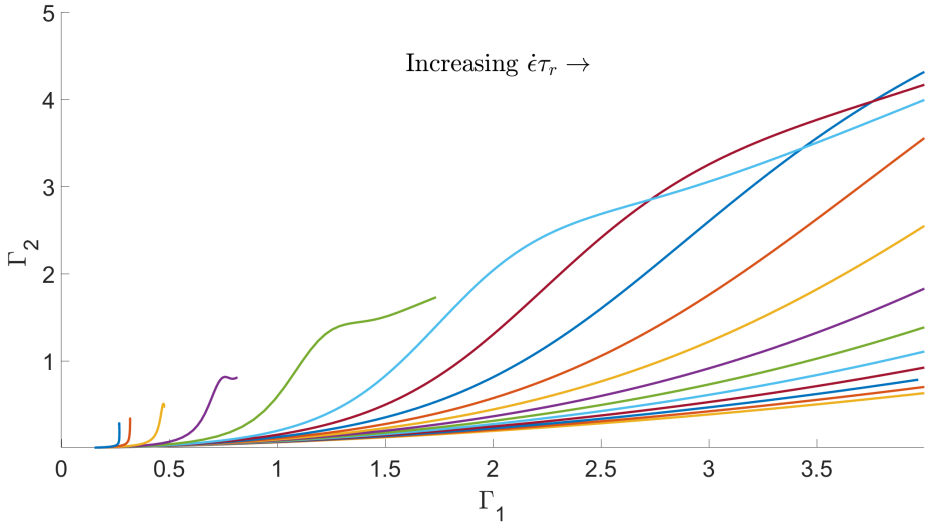


Figure B.1: A plot showing the trajectories of some training data though Γ_i space. From left to right the flow rates displayed are $\dot{\epsilon}\tau_R = [1, 1.1, 1.3, 1.5, 1.7, 1.9, 2, 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3]$.

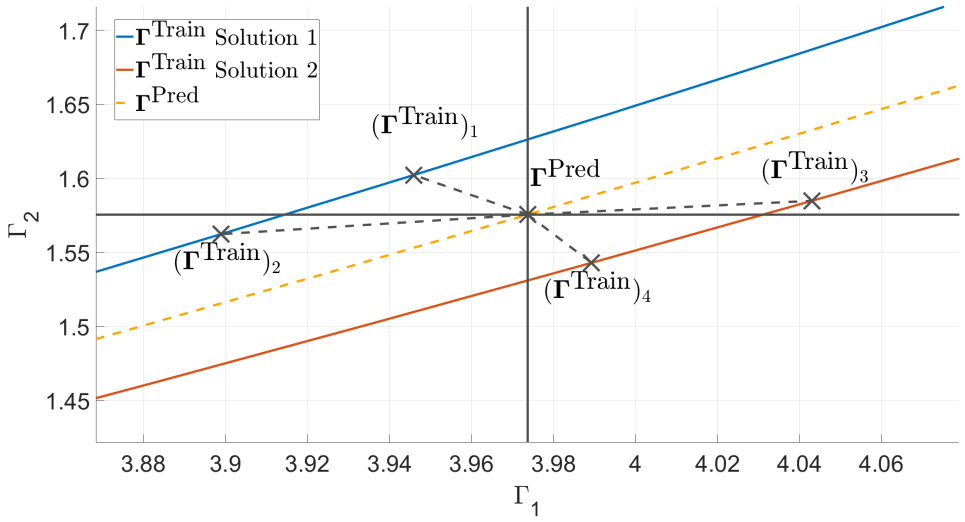


Figure B.2: A Illustration showing the method by which the interpolation points are chosen in quadrant interpolation.

We then extract the full $\mathbf{f}^{\text{Train}}(s, s')$ profile from the training data at each of these four points, and then perform a weighted sum of these profiles based on the relative distances to the current test point. Specifically, we use the following formula for each s and s':

$$\mathbf{f}^{\text{Pred}}(s, s') = C \sum_{q=1}^{4} W(\mathbf{\Gamma}^{\text{Pred}}, \mathbf{\Gamma}_q^{\text{Train}})\ \mathbf{f}_q^{\text{Train}}(s, s'), \tag{B.1}$$

where,

$$C = \frac{1}{\sum_{k=1}^{4} W(\mathbf{\Gamma}^{\text{Pred}}, \mathbf{\Gamma}_k^{\text{Train}})}, \tag{B.2}$$

$\mathbf{f}^{\text{Pred}}$ denotes the interpolation result at a location in $\mathbf{\Gamma}$ phase space with location $\mathbf{\Gamma}^{\text{Pred}}$, $\mathbf{f}_q^{\text{Train}}$ are the profiles used in interpolation from each quadrant with respective locations $\mathbf{\Gamma}_q^{\text{Train}}$ and $W(\mathbf{X}, \mathbf{Y})$ is a normalised weighting function based on the distance in phase space between the location of prediction and the profiles used in the interpolation. Explicitly, the weightings $W(\mathbf{X}, \mathbf{Y})$ are defined as,

$$W(\mathbf{X}, \mathbf{Y}) = \left(\sqrt{\sum_{i=1}^{2} \left(\frac{X_i - Y_i}{\max(\Gamma_i^{\text{Train}})} \right)^2} \right)^{-1}. \tag{B.3}$$

By restricting ourselves to an equal representation of each quadrant we ensure that $\mathbf{f}^{\text{Pred}}$ is a true average of the surrounding data. Without this restriction we could, for example, select all 4 points from the same training trajectory below the test point (x, y), resulting in a prediction that represents a lower value of Γ_2 than the input Γ_2^{Pred}. And thus by using this method, we are able to achieve a relatively fast interpolator that represents the data by choosing a single point from each quadrant.

B.1.2 Gaussian process interpolator

Another option is an interpolator based upon the Gaussian process method [33]. The main difference arises in that if we are to create a Gaussian field with inputs of Γ_1 and Γ_2 then our output is not a single value, but the four-dimensional tensor $f_{\alpha\beta ij}$. This brings into question which elements of $f_{\alpha\beta ij}$ should be considered when optimising the hyperparameters, and this would require further investigation if we wished to apply this method of interpolation. A further difficulty stems from the fact that a Gaussian process type interpolator would consider every point in the data set when estimating, leading to increased calculation times and the possibility of misrepresenting certain quadrants as mentioned in section B.1.1. For example, a test point with a large value of Γ_2^{Pred} may have a far greater number of training data points below it than above it in Γ_i space. To combat these issues, data thinning may need to be employed on areas of Γ_i space with a high density of training points, and care must be taken when selecting a covariance function so that the predictions accurately represent the data.

B.2 3D Interpolation

With the introduction of a third functional, Γ_3, in section 4.3 we require a new interpolator as the dimensions of our $\mathbf{\Gamma}$ phase space have changed. This appendix details or initial octant and radial interpolators in sections B.2.1 and B.2.2 respectively. Our revised radial octant average interpolator which uses elements from both of these two interpolators after both initial interpolators had been proven insufficient is detailed in section B.2.3.

B.2.1 Octant interpolator

The four-point quadrant interpolator from equation (B.1) that we have used throughout two-functional interpolation is not suitable for three functionals, but a natural progression to 3D space is to extend this to eight points, the closest point in each octant. Let us define $\mathbf{\Omega}$ as the domain that contains a vector of coordinates $(\Gamma_1^{\text{Train}}, \Gamma_2^{\text{Train}}, \Gamma_3^{\text{Train}})$ for each point in our training set. If we want to produce a $\mathbf{f}_{\text{Prin}}^{\text{Pred}}$ prediction at $\mathbf{\Gamma}^{\text{Pred}} = (\Gamma_1^{\text{Pred}}, \Gamma_2^{\text{Pred}}, \Gamma_3^{\text{Pred}})$, we use this location to split the region into eight octants. Begin by separating the training data set $\mathbf{\Omega}$ into points with $\Gamma_1^{\text{Train}} \geq \Gamma_1^{\text{Pred}}$ and those without. Repeat this process on both resulting regions for $\Gamma_2^{\text{Train}} \geq \Gamma_2^{\text{Pred}}$ and again on all four regions with $\Gamma_3^{\text{Train}} \geq \Gamma_3^{\text{Pred}}$. Denote these regions as $\mathbf{\Omega}_q(\mathbf{\Gamma}^{\text{Pred}})$ for $q = 1, 2, 3...8$. The formula to interpolate $\mathbf{f}_{\text{Prin}}^{\text{Pred}}$ using the closest training data point to $\mathbf{\Gamma}^{\text{Pred}}$ in each octant is,

$$\mathbf{f}_{\text{Prin}}^{\text{Pred}}(\mathbf{\Gamma}^{\text{Pred}}) = \sum_{q=1}^{8} \begin{cases} W_q \times (\mathbf{f}_{\text{Prin}}^{\text{Train}})_q & \text{for} \quad \mathbf{\Omega}_q(\mathbf{\Gamma}^{\text{Pred}}) \neq \emptyset \\ 0 & \text{for} \quad \mathbf{\Omega}_q(\mathbf{\Gamma}^{\text{Pred}}) = \emptyset \end{cases}, \tag{B.4}$$

where,

$$W_q = \frac{R_q}{R_{Norm}} = \frac{R_q}{\sum_{k=1}^{8} R_k},$$

$$R_q = \begin{cases} \left(\sum_{i=1}^{3} \left(\frac{\Gamma_i^{\text{Pred}} - (\Gamma_i^{\text{Train}})_q}{M_i} \right)^2 \right)^{-\frac{1}{2}} & \text{for} \quad \mathbf{\Omega}_q(\mathbf{\Gamma}^{\text{Pred}}) \neq \emptyset \\ 0 & \text{for} \quad \mathbf{\Omega}_q(\mathbf{\Gamma}^{\text{Pred}}) = \emptyset \end{cases},$$

$(\Gamma_i^{\text{Train}})_q$ denotes the closest training data point in the q^{th} octant with associated value $(\mathbf{f}_{\text{Prin}}^{\text{Train}})_q$ and M_i is a set of scaling parameters explicitly defined in section 4.4.3. Since it is more difficult to produce a set of training data that spans the entire 3D functional phase space, empty octants, $\mathbf{\Omega}_q(\mathbf{\Gamma}^{\text{Pred}}) \neq \emptyset$, occur more frequently than empty quadrants did during 2D interpolation. This can cause interpolation bias, which is discussed in section 4.4.4.

B.2.2 Radial interpolator

Another interpolator we designed for 3D is the radial interpolator. This interpolator takes a weighted average over all points within some distance r up to some maximum number of points, n. We define $\mathbf{\Omega}$ as a set containing every training point's location vector as we did in section B.2.1, then we define $\mathbf{\Omega}_n(\mathbf{\Gamma}^{\text{Pred}})$, a subset which contains the locations of the n closest points. From this we further define $\mathbf{\Omega}_r(\mathbf{\Gamma}^{\text{Pred}})$ as,

$$\mathbf{\Omega}_r(\mathbf{\Gamma}^{\text{Pred}}) = \{\mathbf{x} \in \mathbb{R}^3 \,|\, r > \sqrt{\sum_{i=1}^{3} \left(\frac{\Gamma_i^{\text{Pred}} - x_i}{M_i}\right)^2}, \mathbf{x} \in \mathbf{\Omega}_n(\mathbf{\Gamma}^{\text{Pred}})\},$$

where suitable M_i scalings are discussed in section 4.4.3. The formula to interpolate $\mathbf{f}_{\text{Prin}}^{\text{Pred}}$ using points within some radius r of $\mathbf{\Gamma}^{\text{Pred}}$ is,

$$\mathbf{f}_{\text{Prin}}^{\text{Pred}}(\mathbf{\Gamma}^{\text{Pred}}) = \sum_{p=1}^{n} \begin{cases} W_p \times (\mathbf{f}_{\text{Prin}}^{\text{Train}})_p & \text{for} \quad (\mathbf{\Gamma}^{\text{Train}})_p \in \mathbf{\Omega}_r(\mathbf{\Gamma}^{\text{Pred}}) \\ 0 & \text{for} \quad (\mathbf{\Gamma}^{\text{Train}})_p \notin \mathbf{\Omega}_r(\mathbf{\Gamma}^{\text{Pred}}) \end{cases}, \quad \text{(B.5)}$$

where,

$$W_p = \frac{R_p}{R_{Norm}} = \frac{R_p}{\sum_{k=1}^{n} R_k},$$

$$R_p = \begin{cases} \left(\sum_{i=1}^{3} \left(\frac{\Gamma_i^{\text{Pred}} - (\Gamma_i^{\text{Train}})_p}{M_i}\right)^2\right)^{-\frac{1}{2}} & \text{for} \quad (\mathbf{\Gamma}^{\text{Train}})_p \in \mathbf{\Omega}_r(\mathbf{\Gamma}^{\text{Pred}}) \\ 0 & \text{for} \quad (\mathbf{\Gamma}^{\text{Train}})_p \notin \mathbf{\Omega}_r(\mathbf{\Gamma}^{\text{Pred}}) \end{cases},$$

and the $(\Gamma_i^{\text{Train}})_p$ denotes the location of each point within the set $\mathbf{\Omega}_n(\mathbf{\Gamma}^{\text{Pred}})$ with associated value $(\mathbf{f}_{\text{Prin}}^{\text{Train}})_p$. In practice, this interpolator can produce more accurate results than the octant interpolator when the destiny of training data points in the region $\mathbf{\Omega}_r(\mathbf{\Gamma}^{\text{Pred}})$ is approximately uniform. If the density is sufficiently non-uniform, this interpolator also suffers from interpolation bias.

B.2.3 Radial octant average interpolator

Our initial attempts at predicting shear flows with three functionals were plagued by substantial noise. Investigations into our interpolation method lead us to believe that our octant interpolator was insufficient due to the discrete manner by which it includes or excludes points. Since the full contribution from each octant was entirely dependant on a single point, the reselection of a different point as the prediction evolved could suddenly change the interpolation results from one timestep to the next. This led us to create the radial octant average interpolator.

To interpolate with our radial octant average interpolator, we first choose an octant and find all points within a chosen distance, r, and then repeat this process for each octant. Utilising the notation from section $B.2.1$ with $\mathbf{\Omega}$ being the domain of all training points, and $\mathbf{\Omega}_q(\mathbf{\Gamma}^{\text{Pred}})$ for $q = 1, 2, 3...8$ as the subsets of that domain split into eight separate octants about the prediction location, $\mathbf{\Gamma}^{\text{Pred}}$, we can further define $\mathbf{\Omega}_{rq}(\mathbf{\Gamma}^{\text{Pred}})$ as,

$$\mathbf{\Omega}_{rq}(\mathbf{\Gamma}^{\text{Pred}}) = \{\mathbf{x} \in \mathbb{R}^3 \,|\, r > \sqrt{\sum_{i=1}^{3} \left(\frac{\Gamma_i^{\text{Pred}} - x_i}{M_i} \right)^2}, \mathbf{x} \in \mathbf{\Omega}_q(\mathbf{\Gamma}^{\text{Pred}})\},$$

And then for each subset $\mathbf{\Omega}_{rq}(\mathbf{\Gamma}^{\text{Pred}})$ of the training data we can compute the weighted averages $(\bar{\mathbf{f}}_{\text{Prin}}^{\text{Train}})_q$ and $\bar{R}_q$.

$$(\bar{\mathbf{f}}_{\text{Prin}}^{\text{Train}})_q = \sum_{p=1}^{|\mathbf{\Omega}_{rq}(\mathbf{\Gamma}^{\text{Pred}})|} W_p \times (\mathbf{f}_{\text{Prin}}^{\text{Train}})_p \quad \text{for} \quad (\mathbf{\Gamma}^{\text{Train}})_p \in \mathbf{\Omega}_{rq}(\mathbf{\Gamma}^{\text{Pred}}) \ ,$$

$$\bar{R}_q = \sum_{p=1}^{|\mathbf{\Omega}_{rq}(\mathbf{\Gamma}^{\text{Pred}})|} W_p \times R_p \quad \text{for} \quad (\mathbf{\Gamma}^{\text{Train}})_p \in \mathbf{\Omega}_{rq}(\mathbf{\Gamma}^{\text{Pred}}) \ ,$$

where,

$$W_p = \frac{R_p}{R_{Norm}} = \frac{R_p}{\sum_{k=1}^{|\mathbf{\Omega}_{rq}(\mathbf{\Gamma}^{\text{Pred}})|} R_k} \ ,$$

$$R_p = \Big(\sum_{i=1}^{3} \Big(\frac{\Gamma_i^{\text{Pred}} - (\Gamma_i^{\text{Train}})_p}{M_i} \Big)^2 \Big)^{-\frac{1}{2}} \quad \text{for} \quad (\mathbf{\Gamma}^{\text{Train}})_p \in \mathbf{\Omega}_{rq}(\mathbf{\Gamma}^{\text{Pred}}) \ .$$

These weighted averages create an effective single point in each octant with profile $(\bar{\mathbf{f}}_{\text{Prin}}^{\text{Train}})_q$. We can now use the octant interpolator on these averaged profiles.

$$\mathbf{f}_{\text{Prin}}^{\text{Pred}}(\mathbf{\Gamma}^{\text{Pred}}) = \sum_{q=1}^{8} \begin{cases} \overline{W}_q \times (\bar{\mathbf{f}}_{\text{Prin}}^{\text{Train}})_q & \text{for} \quad \mathbf{\Omega}_{rq}(\mathbf{\Gamma}^{\text{Pred}}) \neq \emptyset \\ 0 & \text{for} \quad \mathbf{\Omega}_{rq}(\mathbf{\Gamma}^{\text{Pred}}) = \emptyset \end{cases} , \tag{B.6}$$

where,

$$\overline{W}_q = \frac{\bar{R}_q}{\bar{R}_{Norm}} = \frac{\bar{R}_q}{\sum_{k=1}^{8} \bar{R}_k}.$$

Since the profiles we use in this version of the octant interpolator are created by averaging many points in each octant, the contribution from a given octant will not drastically change from moment to moment as we evolve $\mathbf{\Gamma}^{\text{Pred}}$. This does happen with our previous octant interpolator which only uses a single point from each octant. As we evolve $\mathbf{\Gamma}^{\text{Pred}}$ which octant certain training profiles are in will change, and thus the result from the individual point interpolator can change significantly with the removal of one point and the addition of another. For this reason, the radial octant average interpolator is far smoother than the octant interpolator during the evolution of $\mathbf{\Gamma}^{\text{Pred}}$ resulting in less noise.

Appendix C

Evaluation of derivatives

After using the GLaMM model, the next step in our original method was to find derivatives for each functional. We can find $\frac{\mathrm{d}\mathbf{S}}{\mathrm{d}t}$ immediately using equation (4.2) and so that becomes step 5. The other functionals however, require us to calculate the derivatives of $\Gamma_1(\mathbf{f}_{\mathrm{Prin}})$ and $\Gamma_2(\mathbf{f}_{\mathrm{Prin}})$, but we do not yet have knowledge of $\frac{\partial}{\partial t}\mathbf{f}_{\mathrm{Prin}}$, and thus we must rotate $\frac{\partial \mathbf{f}}{\partial t}$ into the principal frame of reference. For the choices of Γ_1 and Γ_2 we have used in section 4.1.2,

$$\frac{\mathrm{d}\Gamma_1}{\mathrm{d}t}(t) = \frac{12G_e}{5Z}\int_0^Z \left(\frac{\partial}{\partial t}\mathbf{f}_{\mathrm{Prin}}(s,s,t)\right)_{11} - \left(\frac{\partial}{\partial t}\mathbf{f}_{\mathrm{Prin}}(s,s,t)\right)_{22} ds, \quad \text{(C.1)}$$

and similarly,

$$\frac{\mathrm{d}\Gamma_2}{\mathrm{d}t}(t) = \frac{12G_e}{5Z}\int_0^Z \left(\frac{\partial}{\partial t}\mathbf{f}_{\mathrm{Prin}}(s,Z-s,t)\right)_{11} - \left(\frac{\partial}{\partial t}\mathbf{f}_{\mathrm{Prin}}(s,Z-s,t)\right)_{22} ds. \quad \text{(C.2)}$$

To find $\frac{\partial}{\partial t}\mathbf{f}_{\mathrm{Prin}}$ we begin by using the transformation $\mathbf{f} = \mathbf{V}\cdot\mathbf{f}_{\mathrm{Prin}}\cdot\mathbf{V}^T$, noting that $\mathbf{V}^T = \mathbf{V}^{-1}$ as $\mathbf{V}(t)$ is a tensor composed of orthonormal eigenvectors

due to the symmetry of $\mathbf{S}(t)$. Then, by using the chain rule,

$$\frac{\partial}{\partial t}\mathbf{f}_{\text{Prin}} = \mathbf{V}^T \cdot \mathbf{f} \cdot \frac{\mathrm{d}\mathbf{V}}{\mathrm{d}t} + \left(\mathbf{V}^T \cdot \mathbf{f} \cdot \frac{\mathrm{d}\mathbf{V}}{\mathrm{d}t}\right)^T + \mathbf{V}^T \cdot \frac{\partial \mathbf{f}}{\partial t} \cdot \mathbf{V} \tag{C.3}$$

In order to evaluate these derivatives, we now require a formula for $\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}t}$. To present this analytically, we use an example for when $\sigma_{xy} \neq 0$ in 2D. If we use the notation that S_{ik} denotes each element of $\mathbf{S}$, we have,

$$\mathbf{S} = \begin{pmatrix} S_{xx} & S_{xy} \\ S_{xy} & S_{yy} \end{pmatrix}$$

with eigenvalues,

$$\lambda_1 = \frac{S_{xx} + S_{yy} + \sqrt{S_{xx}^2 + S_{yy}^2 + 4S_{xy}^2 - 2S_{xx}S_{yy}}}{2}$$

$$\lambda_2 = \frac{S_{xx} + S_{yy} - \sqrt{S_{xx}^2 + S_{yy}^2 + 4S_{xy}^2 - 2S_{xx}S_{yy}}}{2}$$

By noting that $\mathbf{S}$ is a symmetric matrix and thus has orthogonal eigenvectors, we write the eigenvectors as,

$$v_1 = \begin{pmatrix} \lambda_1 - S_{yy} \\ S_{xy} \end{pmatrix}, v_2 = \begin{pmatrix} -S_{xy} \\ \lambda_1 - S_{yy} \end{pmatrix}$$

$$\text{or} \quad v_1 = \begin{pmatrix} S_{xy} \\ \lambda_1 - S_{xx} \end{pmatrix}, v_2 = \begin{pmatrix} -\lambda_1 + S_{xx} \\ S_{xy} \end{pmatrix}$$

Normalising these eigenvectors to ensure a orthonormal matrix for $\mathbf{V}$,

$$\mathbf{V} = \frac{1}{\sqrt{(\lambda_1 - S_{yy})^2 + S_{xy}^2}} \begin{pmatrix} \lambda_1 - S_{yy} & -S_{xy} \\ S_{xy} & \lambda_1 - S_{yy} \end{pmatrix}$$

$$\text{or} \quad \frac{1}{\sqrt{(\lambda_1 - S_{xx})^2 + S_{xy}^2}} \begin{pmatrix} S_{xy} & -\lambda_1 + S_{xx} \\ \lambda_1 - S_{xx} & S_{xy} \end{pmatrix}$$

Thus by using S'_{ik} to denote the elements of $\frac{\mathrm{d}\mathbf{S}}{\mathrm{d}t}$,

$$\begin{aligned}\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}t} = \quad & \frac{1}{\sqrt{(\lambda_1 - S_{yy})^2 + S_{xy}^2}} \begin{pmatrix} \lambda'_1 - S'_{yy} & -S'_{xy} \\ S'_{xy} & \lambda'_1 - S'_{yy} \end{pmatrix} \\ & -\frac{(\lambda'_1 - S'_{yy})(\lambda_1 - S_{yy}) + S'_{xy}S_{xy}}{((\lambda_1 - S_{yy})^2 + S_{xy}^2)^{\frac{3}{2}}} \begin{pmatrix} \lambda_1 - S_{yy} & -S_{xy} \\ S_{xy} & \lambda_1 - S_{yy} \end{pmatrix} \\ \text{or} \quad & \frac{1}{\sqrt{(\lambda_1 - S_{xx})^2 + S_{xy}^2}} \begin{pmatrix} S'_{xy} & -\lambda'_1 + S'_{xx} \\ \lambda'_1 - S'_{xx} & S'_{xy} \end{pmatrix} \\ & -\frac{(\lambda'_1 - S'_{xx})(\lambda_1 - S_{xx}) + S'_{xy}S_{xy}}{((\lambda_1 - S_{xx})^2 + S_{xy}^2)^{\frac{3}{2}}} \begin{pmatrix} S_{xy} & -\lambda_1 + S_{xx} \\ \lambda_1 - S_{xx} & S_{xy} \end{pmatrix}\end{aligned}$$

Where,

$$\lambda'_1 = \frac{\mathrm{d}\lambda_1}{\mathrm{d}t} = \frac{1}{2}\left(S'_{xx} + S'_{yy} + \frac{S'_{xx}S_{xx} + S'_{yy}S_{yy} + 4S'_{xy}S_{xy} - S'_{xx}S_{yy} - S'_{yy}S_{xx}}{\sqrt{S_{xx}^2 + S_{yy}^2 + 4S_{xy}^2 - 2S_{xx}S_{yy}}}\right)$$

By rearranging, $\mathbf{V}$ and $\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}t}$ can be written as,

$$\mathbf{V} = \frac{1}{\sqrt{a^2 + b^2}} \begin{pmatrix} a & -b \\ b & a \end{pmatrix} \tag{C.4}$$

$$\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}t} = \frac{a'b - b'a}{(a^2 + b^2)^{\frac{3}{2}}} \begin{pmatrix} b & a \\ -a & b \end{pmatrix} \tag{C.5}$$

$$a = \begin{cases} \lambda_1 - S_{yy} & \text{for} \quad S_{yy} < S_{xx} \\ S_{xy} & \text{for} \quad S_{xx} < S_{yy} \end{cases} \qquad b = \begin{cases} S_{xy} & \text{for} \quad S_{yy} < S_{xx} \\ \lambda_1 - S_{xx} & \text{for} \quad S_{xx} < S_{yy} \end{cases}$$

Where each of S_{ik}, λ_1, a and b only depend on time, t, with no dependance on s or s'. Note that we present two options for $\mathbf{V}$ resulting in two formulas for $\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}t}$. Both of these formulas are equivalent for non-zero shear stress, $\sigma_{xy} \neq 0$, but under purely extensional flow only one formula shall provide

the correct results. This choice is dependant on the principal direction. The first choice shall fail due to a singularity in $\mathbf{V}$ if $S_{xy} = 0$ and $S_{xx} \leq S_{yy}$, whilst the second will not work if $S_{xy} = 0$ and $S_{yy} \leq S_{xx}$. In the case where $S_{xx} - S_{yy} = 0$ and $S_{xy} = 0$ we have degenerate eigenvalues, and neither formula will work.

The trace of $\mathbf{S}$ has a fixed value in polymer dynamics and so the only possible $\mathbf{f}$ that yields a degenerate result is the equilibrium case in 2D, which has no principal direction. For this case when $\mathbf{f} = \mathbf{f}_{eq}$ we can choose $\mathbf{V} = \mathbf{I}$ as no transformation is required between the frames of reference, but we do not know the value of $\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}t}$. This can be alleviated by delaying the start time of the prediction slightly, so that the true value of $\mathbf{f}$ is given for the first time step, and the prediction is not asked to calculate $\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}t}$ using the equilibrium configuration.

Using these formulas, we can now outline how we evaluate the derivatives of Γ_i in the methodology detailed in section 4.1.3. We first calculate the rate of change of the eigenvalues, $\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}t}$, as step 6, since we require this to transform $\frac{\partial \mathbf{f}}{\partial t}$ into $\frac{\partial}{\partial t}\mathbf{f}_{\mathrm{Prin}}$, which becomes step 7. We then proceed as we did in the method without orientation to evaluate derivatives and the evolution of each Γ_i in the $\mathbf{f}_{\mathrm{Prin}}$ frame of reference in step 8.