Reduced Order Models for Non-Newtonian Fluids Using Non-Stationary Gaussian Process Regression

Sven Bendermacher, Vicky Hüfken, Freek Klabbers and Wybe Sesink

Abstract—**This study developed an uncertainty-driven adaptive sampling framework for data-efficient reduced-order models (ROMs) of non-Newtonian fluid flows, leveraging non-stationary Gaussian Process Regression (NSGPR). The objective was to enhance ROMs performance in scenarios with scarce or computationally expensive data, like Finite Element Method (FEM) simulations. We applied Proper Orthogonal Decomposition (POD) to reduce high-dimensional FEM solutions to a set of coefficients, which were then predicted by NSGPR, providing both a mean prediction and an uncertainty measure to guide subsequent adaptive sampling.**

The forward problem demonstrated that NSGPR with adaptive sampling achieved a similar mean absolute error using only one-third of the data compared to linear sampling. For the inverse problem, the framework successfully inferred fluid parameters from limited experimental measurements, yielding accurate posterior distributions which accounted for model and experimental uncertainty.

The method is equation-free and non-intrusive, making it compatible with existing simulation workflows. It provides a promising solution for parameter estimation in nonlinear, data-scarce systems across engineering and physical sciences, significantly reducing the computational burden of high-resolution simulations.

I. Introduction

Simulations and physical experiments often require substantial material or computational resources, making data collection expensive and time-consuming [1]. Reduced-order models (ROMs) provide a way to approximate the data at a fraction of the cost and time [2], [3]. However, building ROMs typically require large amounts of data to train, which defeats the purpose of building a ROM in a data-scarce environment.

One such domain is the modeling of non-Newtonian fluids. Physical experiments are often impractical because of the need to produce and test a wide range of fluids [4]. As a result, numerical methods such as Finite element method (FEM) is used to solve the governing equations for a set of input parameters [5]–[7] . This approach generates a mapping from input parameters to output parameters and is what we refer to as the *forward problem*. Applying this approach systematically across every set of input parameters enables a numerical reconstruction of the relationship between viscosity η and the shear rate $\dot{\gamma}$. Alternatively, the behavior of a non-Newtonian fluid can be learned from inferring experimental observations from a set of fluid parameters [7], [8]. In this case, the goal is to create a mapping from output parameters, for example, the velocity of a ball falling in a fluid, to input parameters, the viscosity of the fluid. This method reverses the mapping and is what we call the *inverse problem*. By combining the forward method and Bayesian inference, the inverse method can be applied to estimate the underlying fluid parameters [9]. Bayesian inference will predict the posterior distribution of the fluid parameters by inferring the experiment data with numerical results using probability distributions. Gaussian Process Regression (GPR) is applied to approximate the forward method and not only provides a prediction of the parameters but also an estimate of the uncertainty of the prediction it has made. Combining Bayesian inference and GPR techniques is relatively new and is currently being applied in engineering applications, an example includes [10].

This project aims to use a novel approach to combine advanced GPR techniques, such as non-stationary kernels, with adaptive sampling techniques to improve Bayesian inference models where little experimental data is available and FEM data is expensive to generate.

The report starts off with the governing equations of the problem and a brief explanation of proper-orthogonal decomposition, non-stationary kernels for Gaussian Process Regression and Bayesian inference, as this is the main knowledge underlying our project. Secondly, the scientific method for the forward and inverse problems will be discussed, followed by a discussion of the numerical and experimental results.

II. Theoretical Background

In this work, we consider a flow around a sphere in a closed cylinder. Body forces act on the sphere such that it moves horizontally. This in turn forces the fluid to flow. We consider a non-Newtonian fluid, i.e. a fluid with viscosity dependent on the shear stress acting upon it.

A. Governing Equations

Consider a flow on some domain Ω with boundary Γ. Denote $\mathbf{u}$ the velocity field on Ω governed by the mass and momentum conservation equations [11]

$$\nabla \cdot \mathbf{u} = 0 \tag{1}$$

and

$$\nabla \cdot 2\eta(\dot{\gamma})\mathbf{D} - \nabla p = \mathbf{0}, \tag{2}$$

where p is the pressure field, $\mathbf{D}$ is the rate-of-deformation tensor and $\eta(\dot{\gamma})$ the viscosity as a function of the shear rate $\dot{\gamma}$. We consider a generalized Newtonian fluid with

Team Internship Report submitted June, 2025; Supervised by Nick Jaensson. Sven Bendermacher (1726803), Vicky Hüfken (1452037), Freek Klabbers (2042274) and Wybe Sesink (1455060) are with the Department of Electrical Engineering, Eindhoven University of Technology. Nick Jaensson is with the Processing and Performance of Materials Group, Eindhoven University of Technology.

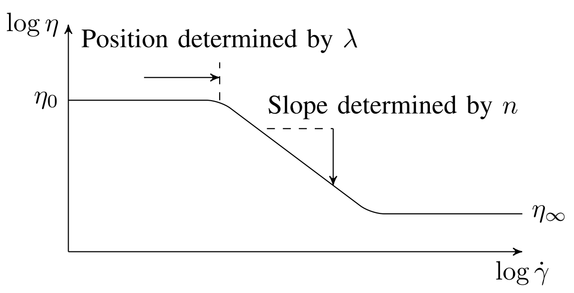


Fig. 1: The effect of relaxation time λ and power law index n on the viscosity versus shear-rate curve in the Carreau viscosity model, shown on a log-log scale.

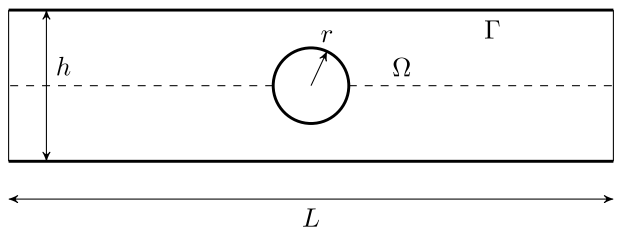


Fig. 2: Schematic of the computational domain Ω with boundary Γ for the falling ball problem within a closed cylinder. The cylinder has a length L and diameter h. The ball has a radius of r.

viscosity according to the Carreau model [12], which is defined as

$$\eta = \eta_\infty + \frac{\eta_0 - \eta_\infty}{(1 + (\lambda\dot\gamma)^2)^{\frac{1-n}{2}}}, \tag{3}$$

where η_0 and η_∞ represent the viscosity at zero and infinite shear rate respectively, λ is the characteristic relaxation time, and n is a parameter related to the slope in the power law region, as shown in Fig. 1. The shear rate is computed through

$$\dot\gamma = \sqrt{2\mathbf{D} : \mathbf{D}}. \tag{4}$$

The domain geometry consists of a cylinder with height h, length L, and symmetry around the horizontal axis, with a sphere of radius r at the origin as shown in Fig. 2.

Denote Γ_w and Γ_s the boundary at the wall and sphere, respectively. The boundary conditions present in the domain are no-slip boundary conditions

$$\mathbf{u} = \mathbf{0} \tag{5}$$

on the walls Γ_w and

$$\int_{\Gamma_s} \boldsymbol{\sigma} \cdot \mathbf{n} dA = \mathbf{F} \tag{6}$$

for the force present on the sphere Γ_s.

Let $w \in \mathbb{W}$ be a scalar field on Ω, with $\mathbb{W}$ the solution space. Here, $w = \|\mathbf{u}\|$ such that w is governed by the mass and momentum equations, and the boundary conditions on Γ. We assume these equations are fully parameterized by some $\theta \in \Theta \subseteq \mathbb{R}^q$, where Θ denotes the parameter space and q is its dimensionality.

Let Ω be discretized into n nodes, such that $\mathbf{w} \in \mathbb{W} \subset \mathbb{R}^n$ is a vector with w_i the solution at node i.

B. Reduced-Order Modeling via POD

We decompose the solution vector into a set of k modes, or basis functions, $\boldsymbol{\phi}_k \in \mathbb{W}$ and coefficients a_k such that we are able to fully construct the solution vector as

$$\mathbf{w}(x) = \sum_k a_k \boldsymbol{\phi}_k(x). \tag{7}$$

The truncated expansion can be written as

$$\mathbf{w}_r^*(x) = \sum_{k=1}^{r} a_k \boldsymbol{\phi}_k(x), \tag{8}$$

where $\mathbf{w}_r^*(x)$ is the reduced model, r is the level of truncation and

$$\mathbb{W}_r = \mathrm{span}(\boldsymbol{\phi}_1, \dots, \boldsymbol{\phi}_r)$$

is the finite-dimensional projection space. Specific to the Reduced Order Model (ROM), $\|\mathbf{w} - \mathbf{w}_r\|$ is minimized for all $\mathbf{w}_r \in \mathbb{W}_r$.

In this work, the set of modes and coefficients are computed through Proper Orthogonal Decomposition (POD). Specific to POD, the ROM uses data to find an orthonormal basis to minimize the error for all truncation levels r. The data that the POD will use is the simulation results from the FEM model of our problem. We prove optimality of the basis generated by POD through Theorem II.1 [13].

Theorem II.1. *Suppose that $\mathbb{W} \subseteq \mathbb{R}^n$ is finite-dimensional and equipped with the standard Euclidean inner product $\langle \mathbf{w}_1, \mathbf{w}_2 \rangle = \mathbf{w}_1^T \mathbf{w}_2$. Store data $\{\mathbf{w}_j \in \mathbb{W} \mid j = 1, \dots, m\}$ as the matrix*

$$\mathbf{W} = (\mathbf{w}_1, \dots, \mathbf{w}_m) \in \mathbb{R}^{n \times m}. \tag{9}$$

Then $\{\boldsymbol{\phi}_k \mid k = 1, \dots, n\}$ is an (orthonormal) POD basis of $\mathbf{W}$ if and only if

$$\mathbf{W}\mathbf{W}^T \boldsymbol{\phi}_k = \lambda_k \boldsymbol{\phi}_k, \tag{10}$$

where $\lambda_1 \geq \dots \geq \lambda_n$, $n = \mathrm{rank}(\mathbf{W})$. Moreover, in that case

$$J(\boldsymbol{\phi}_1, \dots, \boldsymbol{\phi}_r) = \sum_{k>r} \lambda_k \tag{11}$$

From this theorem, we can conclude that the POD basis is obtained from the eigenvalue decomposition of $\mathbf{W}\mathbf{W}^T$. To compute the POD basis, we solve $\mathbf{W}\mathbf{W}^T \boldsymbol{\phi}_k = \lambda_k \boldsymbol{\phi}_k$ through the Singular Value Decomposition (SVD). We write the singular value decomposition

$$\mathbf{W} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T \tag{12}$$

where

$$\mathbf{U} = (\mathbf{u}_1, \dots, \mathbf{u}_N) \in \mathbb{R}^{n \times n},$$
$$\mathbf{V} = (\mathbf{v}_1, \dots, \mathbf{v}_N) \in \mathbb{R}^{m \times m}$$

and

$$\boldsymbol{\Sigma} = \begin{pmatrix} \bar{\boldsymbol{\Sigma}} & 0 \\ 0 & 0 \end{pmatrix}, \quad \bar{\boldsymbol{\Sigma}} = \mathrm{diag}(\sigma_1, \dots, \sigma_n)$$

with $\mathbf{U}$, $\mathbf{V}$ unitary [14]. Then $\mathbf{W}\mathbf{W}^T\mathbf{u}_k = \sigma_k^2\mathbf{u}_k$ and $\lambda_k = \sigma_k^2$. We can conclude that $\boldsymbol{\phi}_k = \mathbf{u}_k$ is the POD basis. $\mathbf{u}_k := \frac{1}{\lambda_k}\mathbf{W}\mathbf{v}_k$ satisfies $\mathbf{W}\mathbf{W}^T\mathbf{u}_k = \lambda_k\mathbf{u}_k$. This minimizes the total error

$$J(\boldsymbol{\phi}_1, \dots, \boldsymbol{\phi}_N) = \sum_{l=1}^{M} \|\mathbf{w}(x, t_l) - \mathbf{w}_r(x, t_l)\|^2 \tag{13}$$

for all truncation levels r. Subsequently,we obtain the POD coefficients through

$$a_k = \boldsymbol{\phi}_k^T \mathbf{w}_k. \tag{14}$$

Using the truncated POD, we can estimate the FEM simulation solutions through the sum of the linear combinations of modes $\boldsymbol{\phi}_k(x)$ and coefficients a_k. As the POD basis $\boldsymbol{\phi}_k$ is already obtained from a few initial simulations, only the POD coefficients are needed to be able to reconstruct the solution field. To obtain a solution field for every set of input parameters in the input parameter space, an accurate prediction of these POD coefficients a_k for unseen parameters is needed.

The goal is to predict the POD coefficients per POD mode for all parameter values in the input parameter space. With this prediction, we want an uncertainty measure so that we can quantify the uncertainty of our model that goes along with this prediction. This uncertainty measure is achieved by applying Gaussian Process Regression, which is discussed in the next section. In Appendix C, the analysis of the POD modes of the problem can be found.

C. Gaussian Process Regression

The model that will be employed is Gaussian process regression (GPR). It is a type of regression that allows for a measure of uncertainty with every prediction. This uncertainty is what makes it viable for a Bayesian inference approach to parameter estimation, since it allows model uncertainty to propagate to parameter uncertainty. The downside of GPR is that with a computational complexity of $\mathcal{O}(N^3)$ [15], [16], it can become computationally expensive for large datasets. Within the context of low-data problems, this should not pose any issues.

GPR works based on a mean function, which contains the estimate, and a covariance function that describes the relationships between points. These are defined as [17],

$$f(\mathbf{x}) \sim \mathrm{GP}(\mu(\mathbf{x}), \Sigma(\mathbf{x}, \mathbf{x}')), \text{with} \tag{15}$$

$$\mu(\mathbf{x}) = \mathbb{E}(f(\mathbf{x})) \tag{16}$$

$$\Sigma(\mathbf{x}, \mathbf{x}') = K(\mathbf{x}, \mathbf{x}') + \mathbf{I}\omega \tag{17}$$

with $f(\mathbf{x})$ being the true function which is approximated with a GP. The GP consists of the mean function $\mu(\mathbf{x})$ and the covariance function $\Sigma(\mathbf{x}, \mathbf{x}')$. The ω term, describes the amount of noise in the measurements. $K(\mathbf{x}, \mathbf{x}')$ is the kernel of the GPR, which determines the local correlation. A common choice is the squared exponential kernel [18],

$$K(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp(-\frac{1}{2l^2}(\mathbf{x} - \mathbf{x}')^2), \tag{18}$$

where l is the length scale, which scales the intensity with which neighboring points are considered. The σ parameter in (18) dictates how much the output value is scaled. The ω term from (17) is the third hyperparameter.

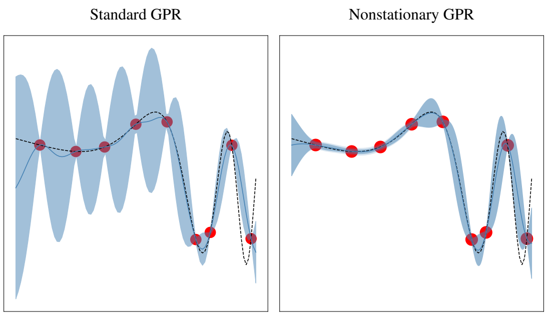


Fig. 3: Comparison between a stationary (left) and non-stationary (right) Gaussian Process (GP). The black dashed line represents the true function. Red dots indicate observed data points. The blue line shows the GP mean prediction, and the shaded blue region denotes standard deviation of predictive uncertainty.

The kernel and these hyperparameters are design choices. There are techniques that optimize GPR hyperparameters [19], [20]. This has one critical issue: in many problems, the properties of the data are not uniform over the entire range. Fig. 3 shows an example of this. The GP learns from the right side of the function that it should have a large variance, and this uncertainty is then extended to the flat left-hand side. This means that a standard GP will perform worse on functions with non-uniform behavior. Since non-Newtonian fluids are a non-linear problem, this is not desirable.

There has been significant research on solving this issue [21]. One solution is to employ non-stationary kernels. These are kernels where the GP parameters are not constant over their entire range. This work considers the approach presented in ”Non-Stationary Gaussian Process Regression with Hamiltonian Monte Carlo” [22].

In this paper, l, σ and ω are seen as latent variables and are determined by using a Gaussian process for each. This leads to the following set of equations:

$$\log(l(\mathbf{x})) \equiv \tilde{l}(\mathbf{x}) \sim \mathrm{GP}(\mu_l, K_l(\mathbf{x}, \mathbf{x}')) \tag{19}$$

$$\log(\sigma(\mathbf{x})) \equiv \tilde{\sigma}(\mathbf{x}) \sim \mathrm{GP}(\mu_\sigma, K_\sigma(\mathbf{x}, \mathbf{x}')) \tag{20}$$

$$\log(\omega(\mathbf{x})) \equiv \tilde{\omega}(\mathbf{x}) \sim \mathrm{GP}(\mu_\omega, K_\omega(\mathbf{x}, \mathbf{x}')) \tag{21}$$

As can be seen in (19), (20) and (21), these latent variables are now determined by a new GP. The kernels that are described in these equations are given by

$$K_c(\mathbf{x}, \mathbf{x}') = \alpha_c^2 \exp(-\frac{1}{2\beta_c^2}(\mathbf{x} - \mathbf{x}')^2), \tag{22}$$

where $c \in \{l, \sigma, \omega\}$. In (22), α_c and β_c are the hyperparameters, corresponding to the variance and length scale, respectively. This means that the full model has nine hyperparameters, $\boldsymbol{\theta} = (\mu_l, \mu_\sigma, \mu_\omega, \alpha_l, \alpha_\sigma, \alpha_\omega, \beta_l, \beta_\sigma, \beta_\omega)$.

The original description of the kernel no longer fits its non-stationary nature introduced above. Instead, the toplevel GP's kernel needs to be changed to the non-stationary generalisation of the squared exponential kernel [23],

$$K_f(\mathbf{x},\mathbf{x}') = \sigma(\mathbf{x})\sigma(\mathbf{x}')\sqrt{\frac{2l(\mathbf{x})l(\mathbf{x}')}{l(\mathbf{x})^2+l(\mathbf{x}')^2}} \times \exp\left(-\frac{(\mathbf{x}-\mathbf{x}')^2}{l(\mathbf{x})^2+l(\mathbf{x}')^2}\right). \tag{23}$$

The non-stationary behaviour is created because l and σ are now dependent on location $\mathbf{x}$. Thus allowing these parameters to change where needed. The non-stationary GP enables the use of adaptive sampling based on uncertainty, since uncertainty now contains more information about its position.

D. Bayesian Inference for Parameter Estimation

Bayes' rule allows us to update our beliefs about parameters $\boldsymbol{\theta}$ using observed data $\mathbf{D}$, and is given by

$$P(\boldsymbol{\theta} \mid \mathbf{D}) = \frac{P(\mathbf{D} \mid \boldsymbol{\theta})P(\boldsymbol{\theta})}{P(\mathbf{D})}, \tag{24}$$

where $P(\boldsymbol{\theta} \mid \mathbf{D})$ is the posterior and the distribution that we wish to know. It represents the probability of a parameter value $\boldsymbol{\theta}$ given the measurement data $\mathbf{D}$. $P(\mathbf{D} \mid \boldsymbol{\theta})$ is the likelihood and is the probability of observing the data given the parameter values $\boldsymbol{\theta}$. $P(\boldsymbol{\theta})$ is the prior belief about the probability distribution of the parameters $\boldsymbol{\theta}$, $P(\mathbf{D})$ is the evidence or marginal probability and shows what the probability is of observing the data $\mathbf{D}$. $P(\mathbf{D})$ is specifically hard to calculate, as it requires integration over the entire parameter space, expressed as

$$P(\mathbf{D}) = \int_M P(\mathbf{D} \mid \boldsymbol{\theta})P(\boldsymbol{\theta})\, d\boldsymbol{\theta}, \tag{25}$$

where M is the space of all the possible parameter values of $\boldsymbol{\theta}$. The high-dimensional nature of M makes this integral often computationally difficult. As a result, this term is suspended as it acts as a normalization and the posterior is calculated without the normalization constant. This reduces Bayes' theorem to:

$$P(\boldsymbol{\theta} \mid \mathbf{D}) \propto P(\mathbf{D} \mid \boldsymbol{\theta})P(\boldsymbol{\theta}). \tag{26}$$

Thus we will obtain a non-normalised probability density function (PDF). The prior distribution can be assumed to be a Gaussian distribution with a prior mean μ_{prior} and standard deviation σ_{prior}. These variables are estimated from the physical problem at hand and allow us to include prior knowledge of the problem. The Gaussian is of the form:

$$P(\boldsymbol{\theta}) = \frac{1}{\sigma_{\text{prior}}\sqrt{2\pi}} \exp\left(\frac{-1}{2}\frac{(\boldsymbol{\theta}-\mu_{\text{prior}})^2}{\sigma^2_{\text{prior}}}\right). \tag{27}$$

The likelihood shows the probability of observing the dataset $\mathbf{D}$, which consists of data points $\mathbf{D} = \{y_1, y_2, \ldots, y_N\}$ given the parameter values $\boldsymbol{\theta}$. The likelihood of observing all the, assuming independent, data points means that it contains the product of the probabilities of each of the data points. The likelihood is the product of Gaussian probabilities that the predictions match the observed data and is of the following form:

TABLE I: 2D parameter space used in the study

Variable	Range	Step Size
n	$0.2 - 1.8$	Linear
λ	$10^{-3} - 10^{5}$	Logarithmic

$$P(\mathbf{D} \mid \boldsymbol{\theta}) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma_{\text{noise}}} \exp\left(\frac{-\|g_i(\boldsymbol{\theta}) - y_i\|^2}{2\sigma^2_{\text{noise}}}\right), \tag{28}$$

where $\|g_i(\boldsymbol{\theta}) - y_i\|^2$ is the squared norm of the difference between the model prediction $g_i(\boldsymbol{\theta})$ of the GPR, based on all parameter values $\boldsymbol{\theta}$, and the experimental observation y_i. Lastly, σ_{noise} defines the noise of observing $\mathbf{D}$, knowing $\boldsymbol{\theta}$. As every data point y_i has different noise level, the noise is rewritten into matrix form,

$$\boldsymbol{\Sigma}_{\text{noise}} = \begin{bmatrix} \sigma_{\text{noise},1} & \cdots & 0 \\ \vdots & \sigma_{\text{noise},i} & \vdots \\ 0 & \cdots & \sigma_{\text{noise},N} \end{bmatrix}. \tag{29}$$

Oftentimes the log-likelihood is used as it results in the summation of Gaussian probabilities. The log-likelihood can be written as [15]:

$$\mathcal{L}(\mathbf{D} \mid \boldsymbol{\theta}) = -n \log\left(\sqrt{2\pi}\boldsymbol{\Sigma}_{\text{noise}}\right) - \frac{1}{2\boldsymbol{\Sigma}^2_{\text{noise}}} \sum_{i=1}^{N} \|g_i(\boldsymbol{\theta}) - y_i\|^2. \tag{30}$$

With the PDF of the prior and the log-likelihood we can analytically determine the posterior distribution, however, this becomes increasingly more complex when there is a high-dimensional parameter space. Therefore we choose to rely on a numerical approach to determine the posterior, namely Markov Chain Monte Carlo (MCMC) for sampling using the Metropolis-Hasting algorithm. This allows us to do numerical Bayesian inference instead of analytical Bayesian inference. The MCMC algorithm will draw samples from the posterior to approximate the quantities such as the mean and standard deviation of the posterior. The specific details of this method are not discussed in this report, as they are outside the scope of this project. The open source package `emcee` is used in this project, for more information, see the paper [24].

III. Method: Forward ROM Problem

In this section, the forward ROM method deployed in our research is expanded upon. The plan of approach and the error estimation is discussed.

In the forward ROM problem, we wish to predict the POD coefficients of unseen input parameters using a Gaussian Process Regression model. The training data will consist of N^2 initial samples obtained from an $N \times N$ linearly-spaced grid in the 2-dimensional input parameter space, consisting of input parameters n and λ. The range of values considered is summarized in Table I.

For data generation, Table II contains a summary of the important parameter values, such as the force F and shear

TABLE II: Summary of parameter description and values for the FEM model of the forward problem.

Description	Symbol	Value
Force on particle	F	1.0
Element size on particle boundary	dx_{part}	0.025
Element size on box boundary	dx_{box}	0.25
Min. element size between particle and wall	$nelem_{\text{min}}$	0.25
Shear viscosity at $\dot{\gamma} \to \infty$	η_∞	10^{-3}
Shear viscosity at $\dot{\gamma} \to 0$	η_0	1.0

viscosity $\dot{\gamma}$ in their limit, that are not changed in the parameter space for the sake of simplicity and the mesh variables used. Appendix B contains a mesh refinement study. The mesh, defined in Table II, was chosen for training the GPR as it balances computation time and low numerical errors, which can be concluded from the mesh refinement study. In the Appendix, Table IV contains all the parameter values for the data generation.

After obtaining the training data, the dataset is reduced to its reduced form by applying POD. This reduction decomposes the solution fields into POD coefficients and modes. The POD modes remain the same for other input parameters; only the POD coefficients will vary. The GPR model is deployed to train on these input parameters and their POD coefficients. For comparison, two GPR models will be trained, one with a stationary kernel and one with a non-stationary kernel. After training the GPR, the model will predict the POD coefficients and corresponding uncertainty for points on the evaluation grid. The evaluation points are the points from a Delaunay triangulation, on which more information can be found below. The evaluated point with the highest uncertainty is where the next data point will be sampled, which is called adaptive sampling, and the results are added to the dataset. Then, the process starts again, the POD is performed on the new dataset and the GPR is trained again. This adaptive sampling approach will be repeated for different numbers of initial samples until the dataset has 100 samples. For each of the models, the mean absolute error (MAE) is calculated to compare the performance of the (non-)stationary GPR and the linear/adaptive sampling technique. The forward problem method is given by the pseudo-code in Algorithm 1.

A. Log-Scale Transformation and Relative Error

A challenge of this ROM is that most of the solutions are three orders of magnitude smaller than the largest value in the dataset. As a result, even minor absolute errors can lead to large relative errors. The relative error is important in this problem because it gives a scale-independent error, better indicating the accuracy of the predicted solution relative to the size.

To address this issue, it is necessary to compress the dynamic range using a logarithmic scale. However, a traditional logarithmic transformation is not feasible for the POD coefficients since they contain both positive and negative values. This necessitates the use of a signed-log scale, which applies a logarithmic transformation to the magnitude while preserving the sign:

$$\tilde{a} = \text{sign}(a) \cdot \log(1 + \|a\|) \tag{31}$$
$$a = \text{sign}(\tilde{a}) \cdot (\exp(\|\tilde{a}\|) - 1). \tag{32}$$

In contrast, for the inverse problem involving the steady-state velocity of the ball, a standard logarithmic transformation is applicable because all velocities point in the same direction.

A secondary challenge emerged with the signed-log transformation. The scale of the uncertainty no longer correctly matched the scale of the POD coefficients. To restore meaningful proportional relationships, the transformed values had to be rescaled such that the ratio of ranges matched the original data. This calibration, $\delta a/\delta\tilde{a}$, allowed uncertainty estimates to remain consistent with the physical interpretation of the model.

Algorithm 1 Pseudocode of forward problem method

```
1: Initialize data grid of $N \times N$ FEM simulations
2: for each iteration do
3:     Perform POD on current snapshots
4:     Transform to reduced space and apply log transform
5:     Train (N)SGP model on parameter inputs and POD coefficients
6:     Evaluate model uncertainty over candidate points (Delaunay)
7:     Select point with highest uncertainty
8:     Run simulation at selected point
9:     Extract results and add to snapshot database
10:    Predict test cases using (N)SGP
11:    Compute and record MAE
12: end for
13: Display final MAE results
```

B. Linear and Adaptive Sampling

Two sampling approaches have been considered: linear and adaptive sampling. For both of these sampling approaches, the initial training data for the GPR is the same. As a baseline, the GPR will be trained on 16 POD coefficients; thus, at least 16 simulation results need to be used. For linear sampling, we consider $n \times n$ grids with $n \in \{4, 5, \ldots, 10\}$, as the maximum number of samples was set at 100. For adaptive sampling a new sample is added to every starting grid until 100 samples have been reached.

Adaptive Sampling is the practice of using the information in previous samples to pick better or more efficient new samples [25]. To sample effectively, a measure of uncertainty can be given for the prediction of the POD coefficient to choose the highest uncertainty as the next sample point. Fig. 4 illustrates this method, where a non-linear test function was sampled, it can be seen that the top-right of the parameters space is sampled more as this shows more varying behavior then the in the rest of the domain. To sample at the highest uncertainty, the

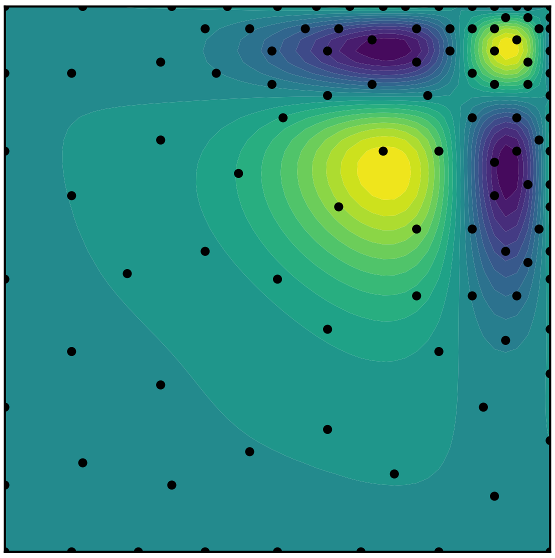


Fig. 4: The mean predicted value of the non-stationary GP after training on the adaptive samples (black dots) of a toy problem. There are more samples in the top-right, where the function changes the most.

uncertainty needs to be known over the whole parameter space, thus the uncertainty has to be evaluated over the whole grid.

Using a conventional, linearly spaced, evaluation grid of $N \times N$ data points to evaluate the uncertainty is a computationally expensive method and scales poorly (N^2 for a 2D parameterspace). This type of evaluation grid is highly dependent on the size of the evaluation grid. Adding more data points makes the sampling too slow, but too few data points either saturate the interesting non-linear regions too quickly or do not properly identify them.

An alternative to a linear evaluation grid was deployed. Delaunay triangulations were implemented to reduce the number of evaluation points and allow for an infinite precision. Delauney triangulation creates a point at the centre of a circle that passes through at least 3 neighbouring points [26]. The created points are in between the current points and on the edge of the parameter space. Thus, generating relevant points to sample the uncertainty. This method also makes sure that more points are created in regions where more points are sampled, fixing saturation issues.

C. Prediction Variance Estimation

The total variance in the forward model is assumed to arise from four sources,

$$\sigma^2 = \sigma^2_{\text{Carreau}} + \sigma^2_{\text{FEM}} + \sigma^2_{\text{POD}} + \sigma^2_{\text{GP}}. \tag{33}$$

Assuming the fluid behaves according to the Carreau model, $\sigma^2_{\text{Carreau}}$ is the variance coming from the Carreau approximation. As this is hard to quantify, it was decided to neglect this term in approximating the error.

The numerical approximation error from the FEM model is σ^2_{FEM}. Since this problem does not have an analytical solution, there is no way of knowing this approximation error. However, the size of the mesh determines the accuracy with which the numerical solution is calculated. We assume that a very fine mesh is our ”ground truth” and determine the approximation error based on this fine mesh. A mesh refinement experiment is done to determine the error of the snapshots generated with a normal mesh w.r.t. to a very fine mesh. By doing this we have an idea of the minimal uncertainty of the whole velocity field due to it being a numerical model with a limited mesh size. The velocity of the ball is the point which is compared in this analysis. The numerical approximation error will be defined as

$$\sigma^2_{\text{FEM}} = (v_{\text{fine}} - v_{\text{forward}})^2, \tag{34}$$

where v_{fine} is the velocity of the ball using the finest mesh and v_{forward} is the velocity of the ball using the mesh for the forward problem.

The error originating for POD truncation is σ^2_{POD}. Since we are using a ROM, we inherently have a POD ROM error. In our analysis, see Subsection III-B, we focused on at most 100 simulations and used only the first 16 POD coefficients for training. Since the error will be the highest when the dataset is the largest, the error is determined for 100 simulations and generalized for all the simulations. To estimate the variance from this truncation, we compute the residual error, using (11), beyond the first 16 modes of a 10×10 linear starting grid using the sum of eigenvalues,

$$\sigma^2_{\text{POD}} = \sum_{k=17}^{100} \lambda_k, \tag{35}$$

where λ_k are the POD eigenvalues and K is the total number of modes.

The predicted uncertainty that the GP provides at every evaluation point is σ^2_{GP}.

D. Quantifying Error

To evaluate the model's predictive performance, we calculate MAE to quantify the difference between the actual velocity field and the predicted velocity field. The MAE will be calculated as

$$\text{MAE} = \frac{1}{N} \sum_{i=1}^{N} |\mathbf{y}_{\text{true}}(\mathbf{x}) - \mathbf{y}_{\text{pred}}(\mathbf{x})|, \tag{36}$$

where $\mathbf{y}_{\text{pred}}(\mathbf{x})$ is the dot product of the predicted POD coefficients $a_{k,\text{pred}}(\mathbf{x})$ and the POD modes of the problem $\phi_k(\mathbf{x})$ and is defined as

$$\mathbf{y}_{\text{pred}}(\mathbf{x}) = \sum_{k=1}^{16} a_{k,\text{pred}}\, \phi_k(\mathbf{x}). \tag{37}$$

IV. Method: Inverse Problem

The inverse problem will enable making a prediction of the viscosity of the fluid based on three data points. These data points are extracted from a real-life experiment where the velocity of an object falling in the liquid is measured. The same experiment is done for different densities of the same object, such that different forces are applied in the same liquid. Then, three different GPs are trained with the same force as in the experiment. However, the GP model

Algorithm 2 Pseudocode of the inverse problem method

```
1: Set up experiment
2: Obtain 3 data points from 3 balls falling through the same liquid
3: Calculate the force on each ball
4: for each force do
5:     Apply log transform
6:     Train (N)SGP model on parameter inputs and velocity
7:     Evaluate model uncertainty over candidate points (Delaunay)
8:     Select point with highest uncertainty
9:     Run simulation at selected point
10:    Extract results and add to snapshot database
11: end for
12: Define prior for Bayesian inference
13: Perform Bayesian inference
14: Display final posterior distribution
```

is trained on the velocity of the ball instead of the whole velocity field, as this is the only data that is obtained from the experiment. This means that for the inverse problem, no POD is applied as it is unnecessary. Bayesian inference, implemented using the `emcee` package [24], has been used to infer the fluid parameters. Lastly, the predicted uncertainty will be compared to the MAE to assess whether the model is underconfident or overconfident. The inverse problem is given by pseudo-code in Algorithm 2.

A. Experimental and Synthetic Data Generation

For the inference problem, an experiment was set up. Three colored balls with different weights were dropped into a cylinder with a non-Newtonian fluid and unknown fluid parameters. Fig. 5 shows the experimental setup.

For every experiment, the falling object was recorded. From these recordings, using the object detection OpenCV library [27], the steady-state velocity of the balls is obtained. At every recorded frame of the recording, the mean y_{pixel}, maximum and minimum pixel values of the object are retrieved. From these, the object's position and diameter are obtained, where the diameter is given by

$$D_{\text{detected}} = y_{\text{max}} - y_{\text{min}}, \tag{38}$$

where D_{detected} is the detected diameter at every time frame, y_{max} is the highest position (in pixels) of the detected coloured object, and y_{min} is the lowest position (in pixels) of the detected coloured object. To reduce the noise in the measurements, the highest and lowest detected object pixels need to be connected with other detected pixels, this is to avoid outliers being used in this formula. The diameter of the ball, D_{ball}, has been measured with a calliper, this information can be used to correct possible camera distortions at each time frame. The pixel position is converted to millimeters using the following method.

$$y_{\text{mm}} = y_{\text{pixel}} \cdot \frac{D_{\text{ball}}}{D_{\text{detected}}} \tag{39}$$

From the position at every time frame, the steady-state velocity is obtained for each experiment as the mean of

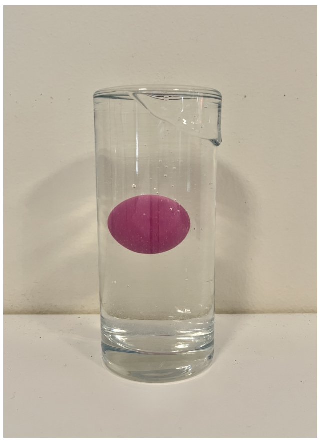


Fig. 5: Experimental setup, a glass filled with soap with a weighted ball sinking to the bottom.

the velocity at each time frame. To know with which forces the GPs need to be trained, the forces acting on the ball are calculated. This consists of 3 components. F_g is the force acting on the object due to Earth's gravity and is defined as

$$F_g = m \cdot g, \tag{40}$$

where m is the mass of the ball and g is the gravitational acceleration. F_b is the buoyancy force acting on the object, which acts in the opposite direction of gravity.

$$F_b = \rho_f \cdot V \cdot g \tag{41}$$

Where ρ_f is the density of the fluid, V is the displaced volume, which corresponds to the volume of the ball when it is fully submerged, and g is the acceleration due to gravity. The density of the fluid is calculated by measuring the weight for a known volume using a measuring cup. Lastly, the last force acting on the ball is the drag, which is computed by the FEM simulation. Combining these three forces, we obtain the net force acting on the ball,

$$F_{\text{net}} = F_g - F_b - F_{\text{drag}}. \tag{42}$$

For the steady-state velocity, the net force acting on the ball equals zero. With this equilibrium the drag force can be found. For each experiment, a GP model is trained with data generated which corresponds to the non-dimensional version of $F_g - F_b$.

The FEM is used to generate synthetic data. The uncertainties for this synthetic data are determined from the real experiment. A benefit of using synthetic data to perform the inverse problem is that the parameters are known, this is not the case for the experimental data. This comparison with the real parameters is nice to show the performance of the model.

B. Bayesian Inference

Using MCMC and adaptive sampling, an estimate of the fluid's viscosity is obtained along with an uncertainty bound. The prior distribution for the Bayesian inference is a uniform distribution over the 2D input parameter space as defined in Table I.

TABLE III: Summary of parameter description and values for the FEM model of the inverse problem.

Description	Symbol	Value
Force on particle	F	1.0
Element size on particle boundary	dx_{part}	0.05
Element size on box boundary	dx_{box}	0.5
Min. element size between particle and wall	$nelem_{\text{min}}$	0.5
Shear viscosity at $\dot{\gamma} \to \infty$	η_∞	10^{-3}
Shear viscosity at $\dot{\gamma} \to 0$	η_0	1.0

C. *Inverse variance Estimation*

The variance estimation consists of the Carreau-model, numerical, experimental uncertainty and the GPs uncertainty. The total variance for the inverse problem consists of four components and is defined as

$$\sigma^2 = \sigma^2_{\text{Carreau}} + \sigma^2_{\text{FEM}} + \sigma^2_{\text{GP}} + \sigma^2_{\text{experiment}}. \tag{43}$$

The Carreau model approximation $\sigma^2_{\text{Carreau}}$ is the same as in the forward problem and is neglected for the same reason. The numerical approximation error originating from the FEM model is σ^2_{FEM}. For the inference problem, a coarser mesh is used. The coarser mesh is double the mesh size of the fine mesh and the most important parameters can be viewed in Table III. The full table with the coarse mesh description can be found in Appendix Table V.

A coarser mesh allows for a faster simulation and a relatively small increase in numerical error, see mesh refinement study in the Appendix B, which is beneficial in the inference problem. Especially since we expect that the numerical error is smaller than the error imposed by the experiment. The numerical error resulting from this mesh is obtained from the same mesh refinement strategy as in the forward problem. The numerical approximation error will be defined as

$$\sigma^2_{\text{FEM}} = (v_{\text{fine}} - v_{\text{inverse}})^2, \tag{44}$$

where v_{fine} is the velocity of the ball using the finest mesh and v_{inverse} is the velocity of the ball using the mesh for the inverse problem.
The predicted uncertainty from the GPR of the velocity of the ball is defined as σ^2_{GP}.

The uncertainty originating from the experiment set-up is $\sigma^2_{\text{experiment}}$. The uncertainty stems from measuring the fluid and object parameters such as the ball diameter and the fluid's density. These all have a capped precision by which they were measured. However, they are neglected as they are very small. The main source of uncertainty comes from measuring the steady-state velocity of the falling ball, due to object detection noise, a range of values for the velocity of the ball is obtained. We define the uncertainty as

$$\sigma^2_{\text{experiment}} = \frac{\sum_i (v_i - \bar{v})^2}{n-1}. \tag{45}$$

The total variance is used to determine the log-likelihood, which is needed for the MCMC algorithm.

D. *Technical Implementation*

Non-Newtonian FEM simulations were performed using TFEM, a TU/e in-house developed finite element toolkit written in modern Fortran. This software was pre-installed in a Docker container, which was called whenever data needed to be sampled.

All other components were implemented in Python 3. The EZyRB library [28] was used for Proper Orthogonal Decomposition (POD) and stationary Gaussian Process regression. The non-stationary Gaussian Process was implemented in-house based on the methodology described in [29], and later enhanced with PyTorch [30] to enable parallel computing. Bayesian inference was carried out using the emcee library [24], and image-based data collection was performed using OpenCV2 [27].

All code written for this project is publicly available in three GitHub repositories. You may find them in section D of the appendix.

V. Results

This section discusses the results of the forward and inverse problems.

A. *Expected velocity field with varying input parameters*

To understand how the variables affect the velocity field, initial data samples were generated for the combinations of variables described in Table I using the FEM. For the variable n, the step size is linear with a size of 0.8, for λ, the steps are taken logarithmically, with each being four orders of magnitude. The results of this initial parameter research are shown in Fig. 6, where the Carreau model is plotted for $n = 0.2, 1.0, 1.8$ and $\lambda = 10^{-3}, 10^1, 10^5$. The red line depicts the viscosity at which the input parameters result.

From this analysis, three cases can be distinguished. When $n = 1$, the fluid behaves as Newtonian, and the viscosity remains constant regardless of the value of λ, consistent with the Carreau model prediction $\eta = \eta_0$. In contrast, for $n < 1$, the viscosity decreases with increasing shear rate, indicating shear-thinning behavior. As a result, the velocity field increases with increasing values of λ. Conversely, when $n > 1$, the fluid displays shear-thickening behavior, where viscosity increases with shear rate, leading to a reduction in the velocity field as λ increases. Also important to note is that for small values of λ the viscosity remains approximately constant, as can be seen in the leftmost column of Fig. 6. This means that the fluid behaves effectively Newtonian, with minimal non-linearity introduced by the viscosity model.

B. *Forward Problem*

1) *Signed-Log Transformation*

The signed-log transformation has a substantial impact on the performance of the model, see Fig. 7. The relative MAE decreased by a factor of 50, for 100 samples using adaptive sampling. The relative error decreased from 164% to 3%, this makes the model more usable in the range where the samples are small. The absolute MAE decreased by a factor of 2, this decrease is welcome but was not the goal of this transformation.

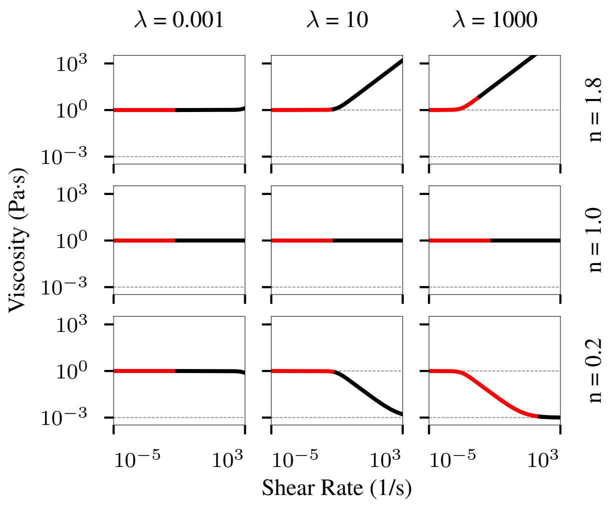


Fig. 6: The Carreau model for selected parameters of $n = 0.2,\ 1.0,\ 1.8$ and $\lambda = 10^{-3},\ 10^{1},\ 10^{5}$. In red is the viscosity shown that is present in the solution of the FEM.

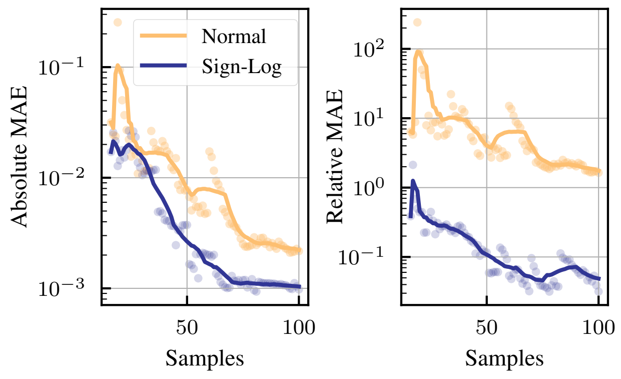


Fig. 7: Comparison of the scaled signed-log transformation (blue) versus the unscaled model (yellow) in terms of error as a function of the number of samples. The left plot shows the absolute MAE, while the right plot shows the relative MAE. Scatter points represent individual errors, and the solid lines are smoothed trends using a Gaussian filter.

2) *Forward Error*

The forward error was defined in (33). Using the mesh for the forward problem, see Table IV, a numerical error of $\sigma^2_{\text{FEM}} = 1.08 \cdot 10^{-07}$ was obtained. Since this error is extremely small it was neglected in the analyses. Secondly, the POD truncation error was determined, see Appendix C, so $\sigma^2_{\text{POD}} = 1.06 \cdot 10^{-05}$. Again, since these are extremely small error ranges, they were neglected, leaving only the GP's uncertainty. Thus although considered, the total error simplifies to $\sigma^2_{\text{forward}} = \sigma^2_{\text{GP}}$.

3) *Comparison of GP Models*

Fig. 8 shows a comparison of the performance of different models and sampling strategies, evaluated in both absolute and relative errors. At a low number of samples, the models are sensitive to small changes, meaning that the addition of a single sample can significantly change the model. To improve the clarity of the trends in adaptively sampled models, a Gaussian filter was applied to the plotted data.

The non-stationary GP with linear sampling has the highest absolute error at low sample counts, which is best explained due to the additional complexity of the model. However, as more samples are added, its performance improves, and it approaches the accuracy of the stationary GP. In the non-scaled problem (i.e., without signed-log transformation), which features a higher degree of non-stationarity, the non-stationary GP with linear sampling outperforms the stationary GP. The best overall performance in terms of absolute error is achieved by the non-stationary GP with adaptive sampling. The adaptively sampled points can be seen in Fig. 10. Even when initialized with a larger linear grid, it quickly decreases to a lower absolute error comparable to a model that started from a smaller initial grid, see Fig. 9. The adaptive sampling techniques acquire the same error at around 35 datapoints as the 10×10 grid. In other words, with adaptive sampling and a non-stationary kernel, the same precision can be reached with one-third the data.

The non-stationary model has a lower relative error compared to the stationary model. The difference in relative error between linear and adaptive sampling is small, a trend that also holds when adaptive sampling is initialized with larger grids, see Fig. 9. For the stationary model with adaptive sampling, the relative error initially increases but decreases after a few samples to a value close to the linear grid sampling. The limited impact of adaptive sampling on relative error is because the samples are chosen based on the highest absolute uncertainty. A sampling strategy based on relative uncertainty could potentially yield better relative error performance, but exploring this is beyond the scope of this study.

C. *Inverse Model*

1) *Inference Error*

The error of our inverse model was defined in (43). Using the inverse problem mesh, see Table V, a numerical error of $\sigma^2_{\text{FEM}} = 2.05 \cdot 10^{-06}$ was obtained. Again, since this is extremely small, it will have very little effect on the uncertainty of the predicted velocity. Therefore, only two terms remain, which are the uncertainty imposed by the experiment and the predicted uncertainty of the GP.

Firstly, the experimental uncertainty arises due to the measurement noise in capturing the velocity of the ball. This uncertainty is determined by the variance of the ball's velocity inside the steady-speed regime. From the recordings, we obtain $\sigma^2_{\text{experiment}} = 7.1 \cdot 10^{-5}$ just based on the uncertainty of the pixel to distance ratio.

Secondly, the predicted uncertainty of the NSGP quantifies the error of the NSGP, but its value is dependent on the amount of data points used and position in the parameter space. Therefore, the precision of the predicted uncertainty was verified. The precision of the predicted uncertainty of the NSGP is determined by looking at the absolute error versus the predicted uncertainty, as seen in Fig. 11. If the predicted uncertainty is accurate, then the absolute error should resemble a half-Gaussian centred at zero, with a width corresponding to the estimated standard deviation, see the red line in Fig. 11. From the distribution of the absolute error of the test samples, it

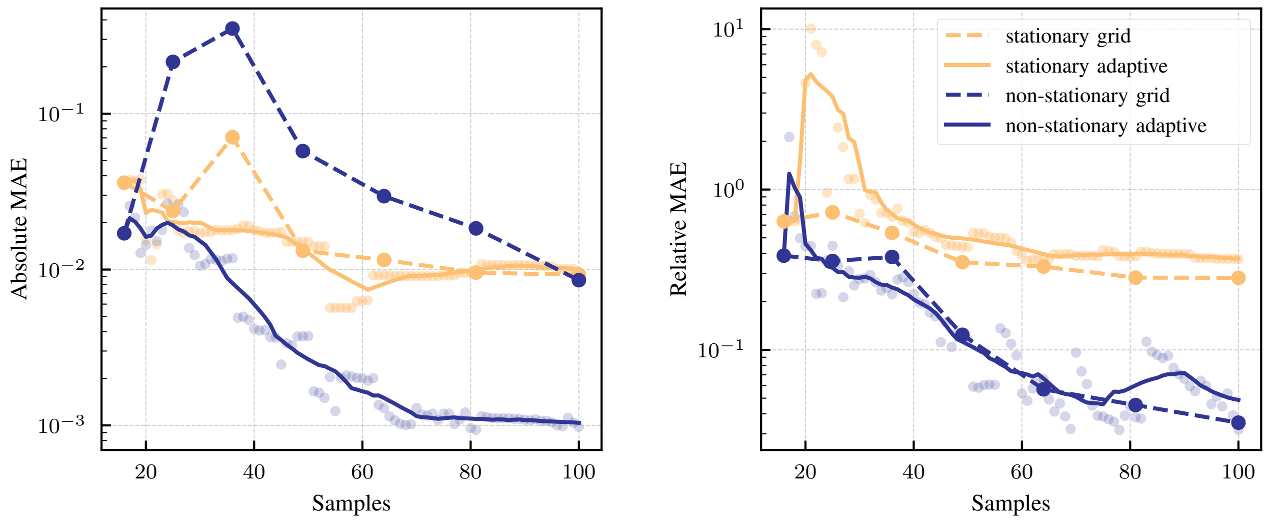


Fig. 8: Comparison of the non-stationary GP (blue) versus the stationary GP (yellow) and grid sampling (dashed lines) versus adaptive sampling (solid lines) in terms of error as a function of the number of samples. The left plot shows the absolute MAE, while the right plot shows the relative MAE. Scatter points represent individual errors, and the solid lines are smoothed trends using a Gaussian filter.

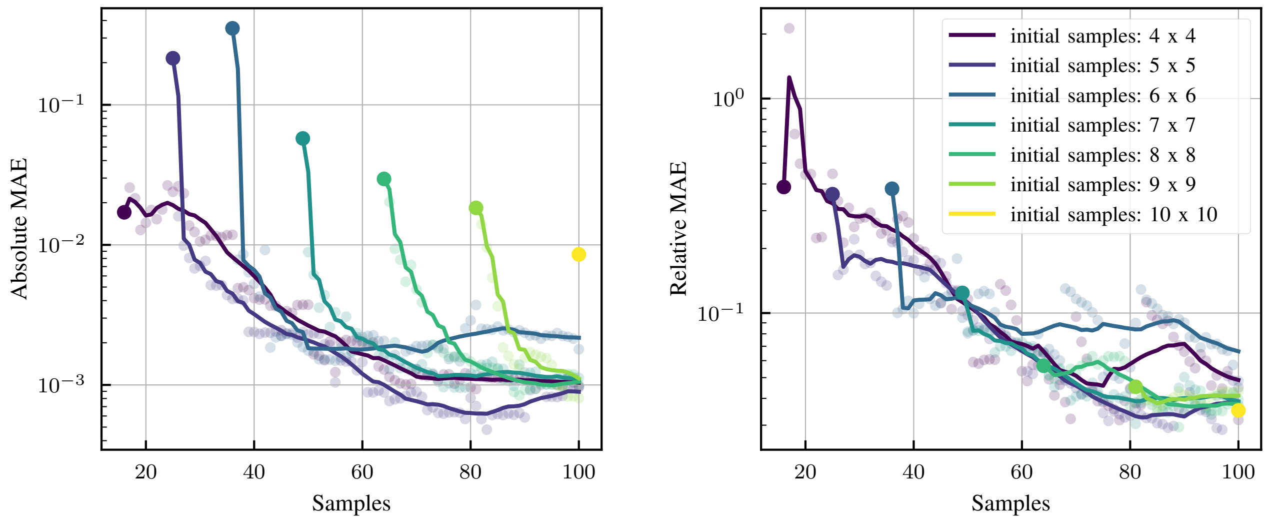


Fig. 9: Comparison of the non-stationary GP performance using different initial sampling grids. The left plot shows the absolute MAE, while the right plot shows the relative MAE. Scatter points represent individual errors, and the solid lines are smoothed trends using a Gaussian filter.

shows that the model is highly under confident. Only one sample passes the 1 σ bound, where it should be 32%. Though this makes overconfident predictions unlikely, it does increase the final uncertainty in the inferred parameters.

2) Inference Results

The measurements of the experiment showed a linear relation between the force and the velocity indicating a Newtonian fluid. To demonstrate the model's capabilities, experiments were simulated using FEM, employing the same uncertainties and forces as the actual experiment. A benefit of using the FEM to produce the data is that it allowed for a comparison with the true value, which would be difficult for real fluids. The inverse problem is performed on three different cases: shear-thinning, Newtonian, and shear-thickening fluids. Each case involves Bayesian inference of the parameters n and λ from three simulated force measurements. No single best point estimate is given of the posteriors, as this choice is application-dependent and outside the scope of this study. Instead, the full posterior distribution is shown to reflect the uncertainty in the inferred parameters.

In the shear-thinning case ($n = 0.5$, $\lambda = 10^4$), the inferred posterior shows a compact distribution that aligns with the true parameter values, see Fig. 12a. The posterior also has some minor peaks at lower λ values, this can be explained by small inaccuracies in the ROM.

In the shear-thickening case ($n = 1.5$, $\lambda = 10^4$), the posterior is wider compared to the shear-thinning scenario, see Fig. 12c. This results from the smaller velocity magnitudes associated with shear-thickening behavior, which, combined with the levels of measurement uncertainty, results in less precise parameter estimation. Despite the spread, the distribution of the posterior aligns well with the true parameter values. Another reason for this large

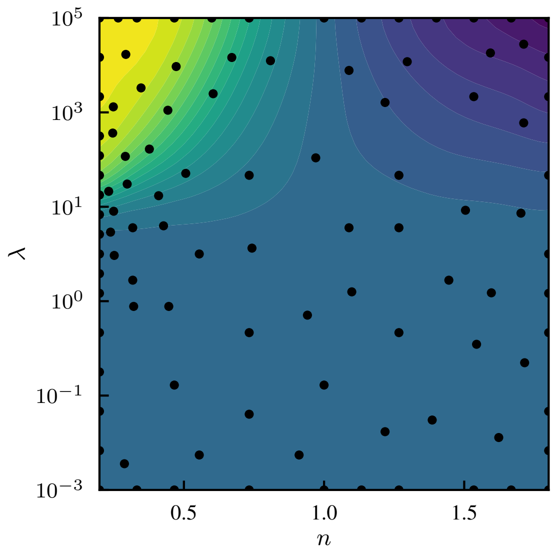


Fig. 10: The mean predicted value of the non-stationary GP after training on the adaptive samples (black dots) of the non-Newtonian problem. There are more samples for low n arround $\lambda = 10^1$, where the function changes the most.

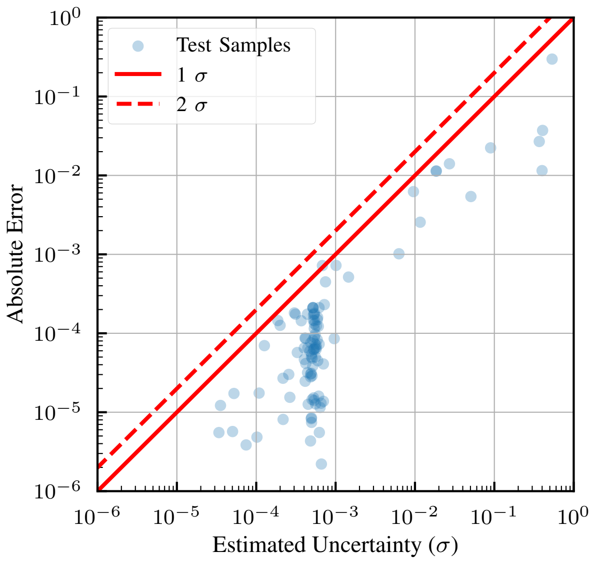


Fig. 11: The absolute error versus the estimated uncertainty of 100 random test points. The red solid line is one standard deviation, and the red dashed line is two standard deviation from mean zero.

spread is that the ROM is under confident as discussed in Subsection V-C1.

For the Newtonian case ($n = 1.0$, $\lambda = 10^0$), the posterior is unable to distinguish between a true Newtonian fluid ($n = 1$) and a non-Newtonian fluid with very small λ, see Fig. 12b. This degeneracy occurs because the Carreau model approaches Newtonian behavior as λ approaches zero, making the two cases nearly identical, as discussed in Subsection V-A. Increasing the number of force measurements would likely reduce the ambiguity, thereby improving inference accuracy and reducing the spread of the distribution.

VI. Discussion

The forward ROM model using a non-stationary GP with adaptive sampling manages to sample efficiently through the parameter space and predicts the POD coefficients better than the other GPs tested. It was demonstrated that the NSGP with adaptive sampling used in the forward ROM problem outperforms the other GPs in one specific problem setup, the sensitivity of the model to changes in this setup has not been investigated, and the accuracy of the model for different domains or governing equations is unknown. Especially the scalability of the model has not been investigated.

While the NSGP with adaptive sampling achieves an improved MAE metric relative to its stationary and linear counterpart, the choice of kernel is one of many, and the choice of the squared exponential kernel has not been sufficiently supported. Optimizations of the NSGP architecture would lead to more optimized adaptive sampling, which in turn aids the prediction returned by the NSGP.

Considering low data $N \leq 100$ the NSGP model with linear sampling performs poorly. While the MAE decreases rapidly with adaptive sampling, the reason for the poor MAE with linear sampling is not sufficiently understood.

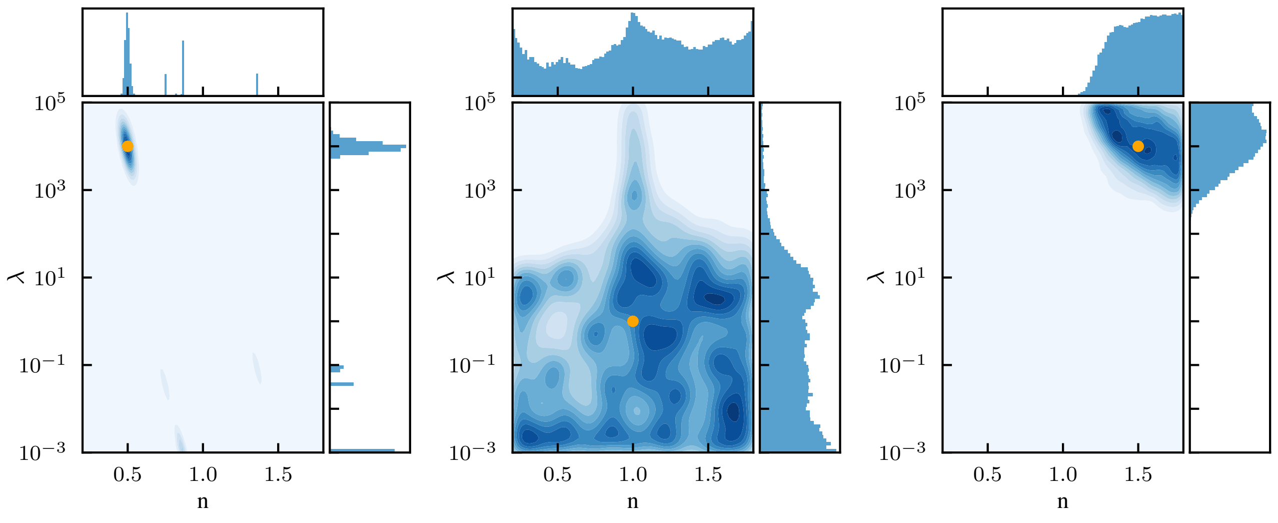
For the inverse problem, the NSGP with adaptive sampling was used in the Bayesian inference. For the inverse problem, there has not been made a comparison between the different GP models and their sampling strategies. It is expected that the NSGP has the best performance as it achieves the lowest error in terms of MAE, but this assumption was never verified and could potentially improve the posterior results. Additionally, it is known that for Bayesian inference the influence of the prior distribution affects the inferred posterior distribution. No other prior distributions were tested, this leaves room for potential future improvements. Lastly, with the inverse problem only fluid parameters are estimated from the measured speed of the ball, a more interesting situation would be to infer the full velocity field.

One of the main aspects of our forward ROM and inverse problem setup, which may severely impact the performance of the model, is the dimensionality of the parameter space. The parametric space considered here consists of two parameters, yet some governing equations may be parameterized by more than two parameters. Both the accuracy of the model and the computational complexity have not been assessed in terms of the parameter space dimensionality.

Additionally, training of the GP is computationally expensive as well, and hence a break-even point exists between the benefit of the model proposed here and existing models. It should be noted that on one side of this point, the time taken for the Gaussian process reduction steps to converge no longer offsets the time taken for numerical simulations to converge.

VII. Conclusion and Recommendations

We have presented a method for sampling and interpolation over parameter spaces in numerical simulations of



(a) (b) (c)

Fig. 12: Joint and marginal posterior distributions inferred from three simulated force measurements using Bayesian inference via MCMC and a reduced-order model (ROM). The orange dots represents the true parameter values. Each subfigure shows a different case: (a) shear-thinning with true parameters $n = 0.5$, $\lambda = 10^4$, (b) Newtonian with true parameters $n = 1.0$, $\lambda = 10^0$, and (c) shear-thickening with true parameters $n = 1.5$, $\lambda = 10^4$.

non-Newtonian fluids. The model takes the form of a non-stationary Gaussian process (NSGP), which yields both an interpolation of proper orthogonal decomposition (POD) modes in the solution space and an uncertainty measure for the reconstruction. We have shown the accuracy of the model inference even for small sample sizes, and its ability to predict the uncertainty in its results effectively. The numerical expense of our model is offset by a significant reduction in the required number of FEM simulations.

The model optimizes coverage of the parameter space through greedy highest-uncertainty sampling while acquiring data for the POD. This adaptive sampling, coupled with the NSGP, results in a decrease in the error over the parameter space and convergence of the solution. Adaptive sampling was benchmarked against linear grid sampling, which demonstrates a sharp improvement in convergence in favour of the adaptive method. It has been shown that in our case, only a third of the data was needed to gain the same accuracy.

In addition to the forward problem, the method was successfully extended to an inverse problem using Bayesian inference. By leveraging NSGP predictions and experimental data, the method provided accurate posterior distributions over fluid parameters, including the NSGP's model uncertainty. This demonstrates the viability of the approach in real-world parameter estimation tasks under data-scarce conditions. Though this method is currently under-confident, this could be calibrated to better match the accuracy.

An attractive characteristic of this method is that it is equation-free, so no prior information on the governing PDE is required. The greedy algorithm is non-intrusive, and as a consequence, can be combined with existing numerical simulation pipelines.

This approach is promising for other non-linear and computationally expensive FEM models, though this approach has only been tested on this one case. For problems with a more stationary field, an NSGP will likely not outperform their stationary counterpart.

Future work may explore generalization to higher-dimensional parameter spaces, alternative kernel formulations, adaptive sampling strategies based on relative uncertainty and inferring full velocity fields instead of parameter values. Expanding experimental validation across more fields of study would also strengthen the applicability of this framework.

References

[1] T. Lassila, A. Manzoni, A. Quarteroni, and G. Rozza, "Model order reduction in fluid dynamics: Challenges and perspectives," in *Reduced Order Methods for Modeling and Computational Reduction*, ser. MS&A - Modeling, Simulation and Applications, Q. A and R. G, Eds., vol. 9, Cham, Germany: Springer, 2014, pp. 235–273, ISBN: 978-3-319-02089-1. DOI: 10.1007/978-3-319-02090-7_9.

[2] J. S. H. Mengwu Guo, "Data-driven reduced order modeling for time-dependent problems," *Computer Methods in Applied Mechanics and Engineering*, 2018.

[3] B. Kulp. "What is a reduced order model and what's its product development role?" (), URL: https://www.ansys.com/blog/what-is-a-reduced-order-model-response-surface-model (visited on 03/08/2025).

[4] R. P. Chhabra, "Non-newtonian fluids: An introduction," in *Rheology of Complex Fluids*, A. P. Deshpande, J. M. Krishnan, and P. B. S. Kumar, Eds., Springer, 2010, pp. 3–34.

[5] H.-C. Huang, Z.-H. Li, and A. S. Usmani, *Finite Element Analysis of Non-Newtonian Flow, Theory and Software*, 1st ed. London: Springer London, 1999, 218 pp., Published as part of the Springer Book Archive, ISBN: 978-1-4471-1204-4. DOI: 10.1007/978-1-4471-0799-6.

[6] S. Direct. "Finite element method." (2021), URL: https://www.sciencedirect.com/topics/engineering/finite-element-method (visited on 03/08/2025).

[7] A. Fabbri and C. Cevoli, "Rheological parameters estimation of non-newtonian food fluids by finite elements model inversion," *Journal of Food Engineering*, vol. 169, pp. 172–178, 2016, ISSN: 0260-8774. DOI: https://doi.org/10.1016/j.jfoodeng.2015.08.035. URL: https://www.sciencedirect.com/science/article/pii/S026087741500388X.

[8] Y. I. Cho, J. P. Hartnett, and W. Y. Lee, "Non-newtonian viscosity measurements in the intermediate shear rate range with the falling-ball viscometer," *Journal of Non-Newtonian Fluid Mechanics*, vol. 15, pp. 61–74, 1984, Received August 2, 1983.

[9] A. Kontogiannis, R. Hodgkinson, S. Reynolds, and E. L. Manchester, "Learning rheological parameters of non-newtonian fluids from velocimetry data," Under consideration for publication in *J. Fluid Mech.*, Nov. 15, 2024. arXiv: 2408.02604 [physics.flu-dyn]. URL: https://arxiv.org/abs/2408.02604v2.

[10] R. Willems, P. Förster, S. Schöps, O. van der Sluis, and C. V. Verhoosel, "A probabilistic reduced-order modeling framework for patient-specific cardio-mechanical analysis," *Computers in Biology and Medicine*, vol. 190, p. 109 983, 2025, ISSN: 0010-4825. DOI: https://doi.org/10.1016/j.compbiomed.2025.109983. URL: https://www.sciencedirect.com/science/article/pii/S0010482525003348.

[11] R. B. Bird, R. C. Armstrong, and O. Hassager, *Dynamics of Polymeric Liquids: Volume 1, Fluid Mechanics*, 2nd ed. New York: Wiley-Interscience, 1997, ISBN: 978-0-471-57495-4.

[12] P. J. Carreau, "Rheological equations from molecular network theories," *Transactions of The Society of Rheology*, vol. 16, no. 1, pp. 99–127, Mar. 1972, ISSN: 0038-0032. DOI: 10.1122/1.549276. eprint: https://pubs.aip.org/sor/jor/article-pdf/16/1/99/12754934/1_549276.pdf. URL: https://doi.org/10.1122/1.549276.

[13] P. Benner, A. Cohen, M. Ohlberger, and K. Willcox, Eds., *Model Reduction and Approximation: Theory and Algorithms* (Computational Science and Engineering). Philadelphia, PA: Society for Industrial and Applied Mathematics, 2017, vol. 15, ISBN: 978-1-61197-481-2. DOI: 10.1137/1.9781611974829. URL: https://epubs.siam.org/doi/book/10.1137/1.9781611974829.

[14] S. L. Brunton and J. N. Kutz, *Data Driven Science & Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge, UK: Cambridge University Press, 2019.

[15] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[16] D. K. Duvenaud, "Automatic model construction with gaussian processes," PhD dissertation, Pembroke College, Cambridge, UK, Jun. 2014. URL: https://www.cs.toronto.edu/~duvenaud/thesis.pdf.

[17] Y. Shi. "Gaussian processes, not quite for dummies." (2019), URL: https://thegradient.pub/gaussian-process-not-quite-for-dummies/ (visited on 03/08/2025).

[18] D. Duvenaud. "The kernel cookbook." (2014), URL: https://www.cs.toronto.edu/~duvenaud/cookbook/ (visited on 05/29/2025).

[19] S. Manzhos and M. Ihara, "On the optimization of hyperparameters in gaussian process regression with the help of low-order high-dimensional model representation," *arXiv preprint arXiv: 2112.01374*, 2021. URL: https://arxiv.org/pdf/2112.01374.

[20] S. Winarni and S. W. Indratno, "Gaussian process regression with average hyperparameter," *Journal of Physics: Conference Series*, vol. 1496, no. 1, Mar. 2020. URL: https://www.proquest.com/scholarly-journals/gaussian-process-regression-with-average/docview/2569764647/se-2.

[21] A. S. Booth, A. Cooper, and R. B. Gramacy, "Nonstationary gaussian process surrogates," arXiv preprint arXiv:2305.19242v2 [stat.ME], Dec. 2024. URL: https://arxiv.org/abs/2305.19242v2.

[22] M. Heinonen, H. Mannerström, J. Rousu, S. Kaski, and H. Lähdesmäki, "Non-stationary gaussian process regression with hamiltonian monte carlo," in *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, Available at https://arxiv.org/pdf/1508.04319, vol. 51, Cadiz, Spain: PMLR, May 2016, pp. 732–740. URL: https://proceedings.mlr.press/v51/heinonen16.html.

[23] M. N. Gibbs, "Bayesian gaussian processes for regression and classification," Ph.D. dissertation, University of Cambridge, 1997. URL: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=b5a0c62c8d7cf51137bfb079947b8393c00ed169.

[24] D. Foreman-Mackey, D. W. Hogg, D. Lang, and J. Goodman, "Emcee: The mcmc hammer," *Publications of the Astronomical Society of the Pacific*, vol. 125, no. 925, pp. 306–312, Mar. 2013, ISSN: 1538-3873. DOI: 10.1086/670067. URL: http://dx.doi.org/10.1086/670067.

[25] Q. F. Stout and J. Hardwick, *Adaptive sampling designs*, Accessed: 2025-04-28, 1997–2017. URL: https://web.eecs.umich.edu/~qstout/AdaptSample.html.

[26] I. Henry. "Visualizing delaunay triangulation." (Jul. 2022), URL: https://ianthehenry.com/posts/delaunay/ (visited on 04/30/2025).

[27] O. Team, *Opencv: Open source computer vision library – python bindings*, https://github.com/opencv/opencv-python, Version 4.11, 2025. (visited on 06/01/2025).

[28] N. Demo, M. Tezzele, and G. Rozza, "EZyRB: Easy Reduced Basis method," *The Journal of Open Source Software*, vol. 3, no. 24, p. 661, 2018. DOI: https://doi.org/10.21105/joss.00661.

[29] M. Heinonen, H. Mannerström, J. Rousu, S. Kaski, and H. Lähdesmäki, "Non-stationary gaussian process regression with hamiltonian monte carlo," *Artificial Intelligence and Statistics*, pp. 732–740, 2016.

[30] J. Ansel, E. Yang, H. He, *et al.*, "PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation," in *29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2 (ASPLOS '24)*, ACM, Apr. 2024. DOI: 10.1145/3620665.3640366. URL: https://docs.pytorch.org/assets/pytorch2-2.pdf.

[31] OpenAI. "Hello gpt4-o." (2024), URL: https://openai.com/index/hello-gpt-4o/ (visited on 02/17/2025).

Declaration of AI Tools

AI tools have been used in various parts of this project and the writing of the report. This section aims to declare all AI tools used in both the report and other work done. The authors hereby declare that all AI output has been checked and take full responsibility for all content delivered, both in the report and in the code.

Writing tools used

- ChatGPT-4o and ChatGPT-4o mini [31] have both been used to format certain Biblatex references.
- Grammarly has been used; this is a spelling and grammar check on texts. It checks written text in real time and rewrites small portions, but never more than a sentence.

Coding tools used

- GitHub Co-pilot is a coding AI that can be installed in Visual Studio Code which has access to the entire code base (if the user so chooses). It has been used to help debug code and rewrite sections to be more concise.

Appendix

A. *FEM settings*

Table IV shows the FEM settings used for the simulations of the forward problem.

TABLE IV: Parameter description and values for the FEM Model of the forward problem.

Variable name	Value
Force on Particle F	1.0
Type of Particle: Sphere	T
Radius Sphere: radr	1.0
Radius cylinder: radc	2.0
Length of the enclosing cylinder (left): lenc1	10.0
Length of the enclosing cylinder (right): lenc2	10.0
Element size on particle boundary: dx_{part}	0.025
Element size on the box boundary: dx_{box}	0.25
Minimal element size between particle and wall: $nelem_{min}$	0.25
Generalized Newtonian model	2:Carreau
Yield model	0: No yield function
Shear viscosity at infinite shear rate: η_∞	$10e-3$
Shear viscosity at zero shear rate: η_0	1.0
Transition control factor: a	2.0

Table IV shows the FEM settings used for the simulations of the inverse problem.

TABLE V: Parameter description and values for the FEM Model of the inverse problem.

Variable name	Value
Force on Particle F	1.0
Type of Particle: Sphere	T
Radius Sphere: radr	1.0
Radius cylinder: radc	2.0
Length of the enclosing cylinder (left): lenc1	10.0
Length of the enclosing cylinder (right): lenc2	10.0
Element size on particle boundary: dx_{part}	0.050
Element size on the box boundary: dx_{box}	0.50
Minimal element size between particle and wall: $nelem_{min}$	0.50
Generalized Newtonian model	2:Carreau
Yield model	0: No yield function
Shear viscosity at infinite shear rate: η_∞	$10e-3$
Shear viscosity at zero shear rate: η_0	1.0
Transition control factor: a	2.0

B. *FEM error*

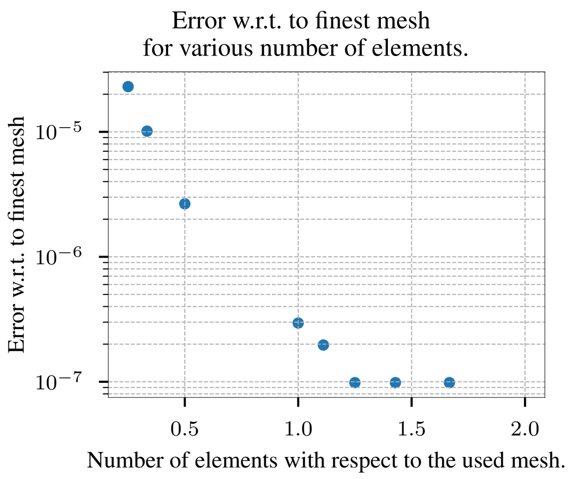
A mesh refinement experiment was done to determine the error of the snapshots generated with a normal mesh w.r.t. to a very fine mesh. By doing this we have an idea of the minimal uncertainty of the whole velocity field due to it being a numerical model with a limited mesh size. The velocity of the ball is the point which was compared in this analysis. Furthermore, the mesh refinement was performed with the parameters $n = 0.2$ with $\lambda = 1e5$ and $n = 1.8$ with $\lambda = 1e5$, since, from the initial parameter study V-A, we know it is the most varying in terms of the viscosity. The results are discussed in Table VII and Figure 14. We can see that the numerical approximation error of the forward mesh (number of elements equals 1 and described in Table IV) is of order 10^{-7} w.r.t. to the finest mesh that was calculated (number of elements equals 2) for $n = 0.2$. For $n = 1.8$, the relative error of the mesh used is of order $3 \cdot 10^{-7}$. These are acceptable approximation errors as the simulation time quickly increases with a finer mesh. Taking the maximum approximation error we can conclude that for the forward problem, $\sigma^2_{\text{FEM}} = 1.0802744e-07$. If we double the mesh size and obtain 0.5 elements w.r.t. to forward problem mesh, the relative error is almost 20 times bigger, but the computation is 10 times faster. For $n = 0.2$, the relative error is 10 times greater and the computation time is 8 times faster. Therefore, it was decided that this mesh is suitable for the inverse problem, as fast computation time is needed. Taking the maximum approximation error we can conclude that for the inverse problem, $\sigma^2_{\text{FEM}} = 2.0525215e-06$

TABLE VI: Velocity of ball for different mesh sizes at $n = 0.2$ and $\lambda = 1e5$.

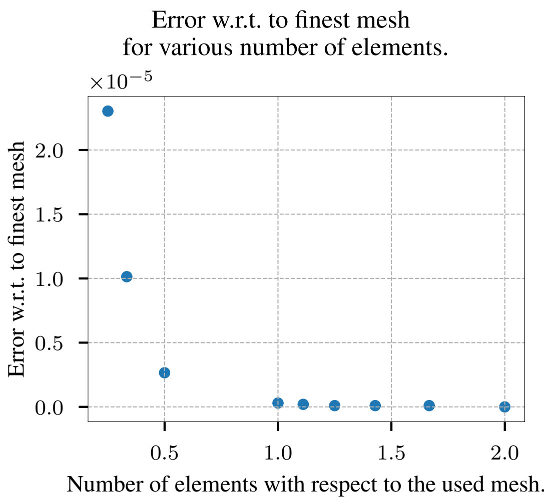
# of elements w.r.t. mesh of forward problem	Velocity at the sphere	Relative error	simulation time
0.25	8.82778644561767578125	3.2624288e-05	4.5s
0.33	8.82798767089843750000	9.830497e-06	7.8s
0.5	8.82805633544921875000	2.0525215e-06	20s
1	8.82807350158691406250	1.0802744e-07	3m30s
$\frac{10}{9}$	8.82807350158691406250	1.0802744e-07	6m
$\frac{10}{8}$	8.82807445526123046875	0.0	10m
$\frac{10}{7}$	8.82807445526123046875	0.0	20m
$\frac{10}{6}$	8.82807445526123046875	0.0	40m
2	8.82807445526123046875	0.0	45m

TABLE VII: Velocity of ball for different mesh sizes at $n = 1.8$ and $\lambda = 1e5$.

# of elements w.r.t. mesh of forward problem	Velocity at the sphere	Relative error	simulation time
0.25	2.9576072120107710361480712890625e-04	2.3025841e-05	6.0s
0.33	2.9576453380286693572998046875000e-04	1.0135306e-05	9.6s
0.5	2.9576674569398164749145507812500e-04	2.656828e-06	40s
1	2.9576744418591260910034179687500e-04	2.952031e-07	5m14s
$\frac{10}{9}$	2.9576747328974306583404541015625e-04	1.9680206e-07	7m
$\frac{10}{8}$	2.9576750239357352256774902343750e-04	9.840103e-08	12m
$\frac{10}{7}$	2.9576750239357352256774902343750e-04	9.840103e-08	17m
$\frac{10}{6}$	2.9576750239357352256774902343750e-04	9.840103e-08	32m
2	2.9576753149740397930145263671875e-04	0.0	2hr

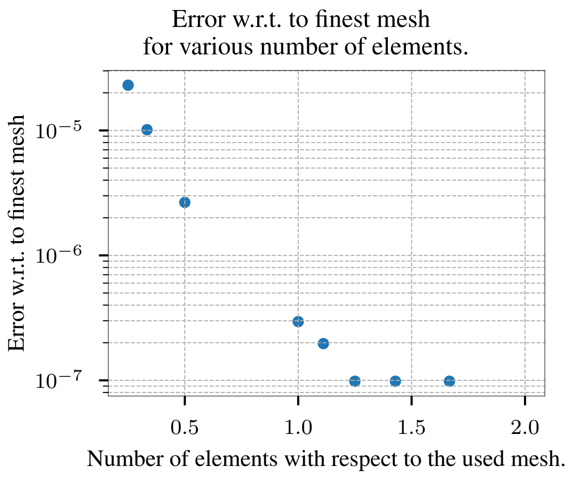


(a) Logarithmic scale

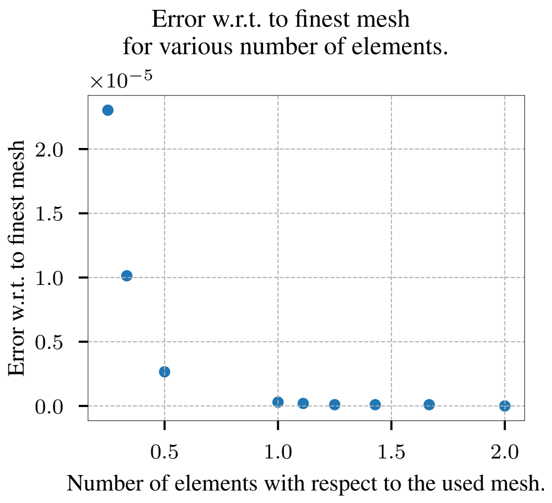


(b) Linear scale

Fig. 13: Approximation error w.r.t. current mesh size on logarithmic and normal scale for $n = 1.8$.



(a) Logarithmic scale



(b) Linear scale

Fig. 14: Approximation error w.r.t. mesh size for the forward problem on logarithmic and linear scale for $n = 0.2$.

C. POD approximation error

Starting with 100 simulations, scattered as a 10x10 linear grid in the parameter space, we can obtain the following results for the energy and the cumulative energy every mode contains, see Figure 15 and 16. We can observe that we have the same number of POD modes as we have data points, which is as expected. However, we can accurately reconstruct the solution of a simulation by combining the velocity fields of the strongest modes and the correct POD coefficients. From the cumulative sum, we know that the first 16 modes contain 99.99961% of the information, this means that the truncation has 0.0011205673% of the information. The GPR is trained on the first 16 coefficients that correspond to the first 16 modes. From equation 11, we know that the total error is bounded by the sum of the eigenvalues values not included in the ROM. From this we obtain $\sigma^2_{\text{POD}} = 1.0580309e-05$.

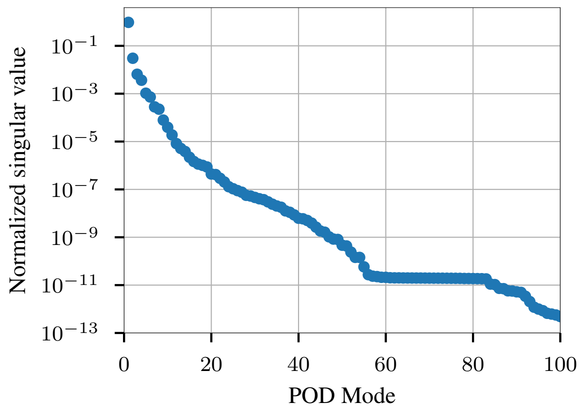


Fig. 15: The energy captured in every POD mode.

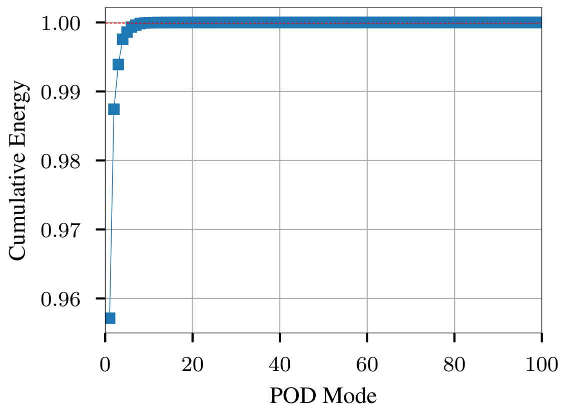


Fig. 16: The cumulative energy captured in every POD mode.

Fig. 17: Eigenvalue analysis for 100 simulations in a 10×10 linear grid.

Applying the POD ROM, we can also achieve the POD coefficients for every POD mode. Figure 18 shows the POD coefficients per POD mode for every set of input parameters used to generate the data using a 4×4 grid. We can clearly see that the solution behaves non-linear, the POD coefficients change the most for low values of n and high values of λ. Additionally, we can see that the coefficients become smaller per POD mode, confirming that the first POD modes are the most important and contain the most information. The first 16 POD coefficients for every simulation will be used to train the GPR model.

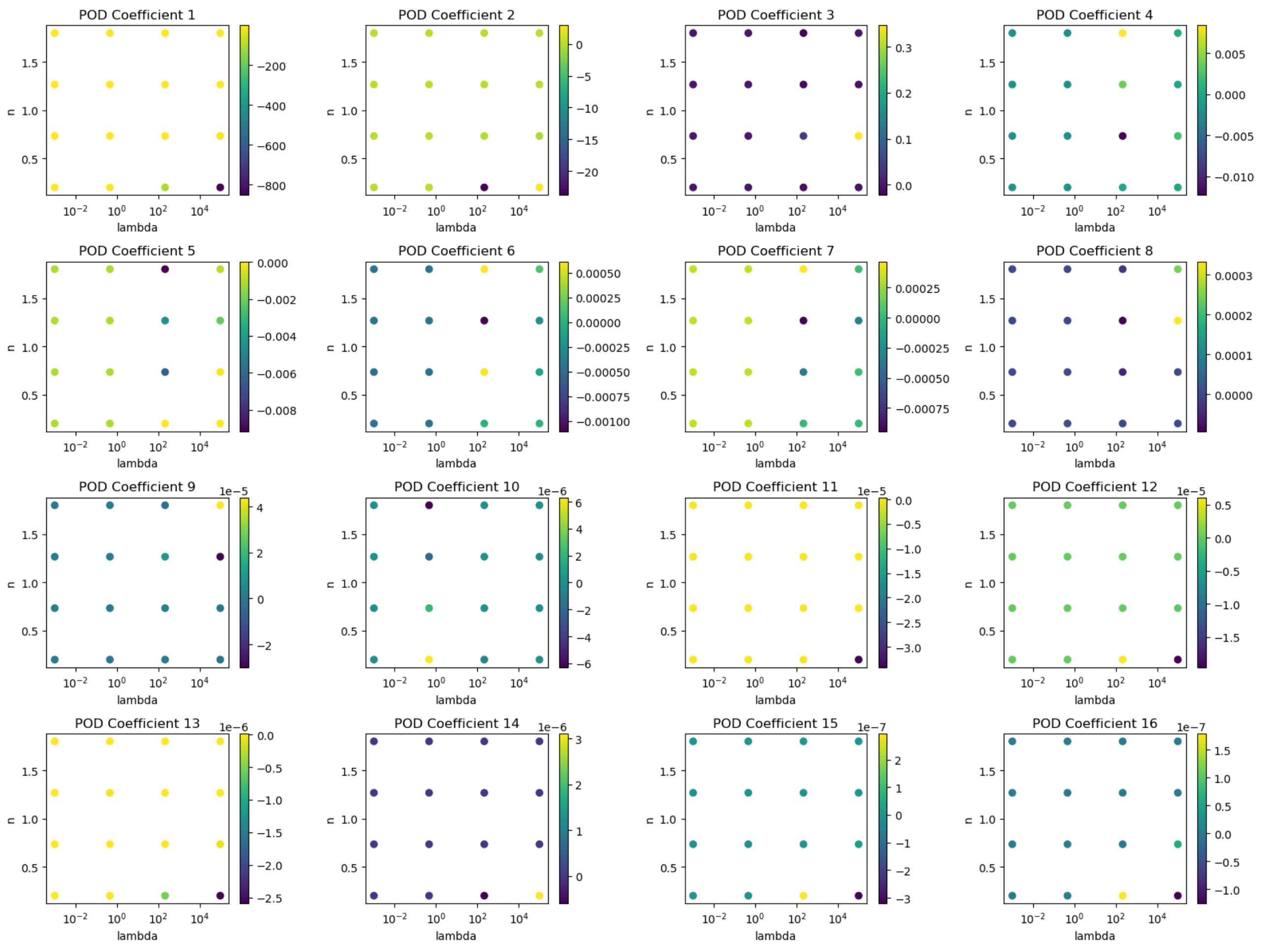


Fig. 18: POD coefficients for 4x4 grid per POD mode.

D. Code & GitHub Repos

This subsection of the appendix contains links to the relevant code made during this project. This code was divided into a few repositories. Below each repositories will briefly be expanded upon.

1) Non-Stationary Gaussian Process

The NSGP repository (https://github.com/Frakert/NSGPR) contains a Python library for using the stationary Gaussian process. It is written using Pytorch, allowing for GPU acceleration and parallel processing. It can be installed like any other Python package and uses similar methods to sklearn. The repository contains a few examples showing how the package works. This is likely the most widely applicable piece of code from this project.

2) Fortran-Bridge

The Fortran-Docker Bridge repository (https://github.com/Frakert/Fortran_Docker_Bridge) contains a Python library for calling a Docker container that contains an installation of TFEM. The package was build to encapsulate the initialization of the Docker container, and to be able to call it for samples. This makes it such that the Docker container can be regarded as a black box. This Package is highly specific to our use case, and is thus likely not very useful outside a few TFEM users.

3) ROM-NNFM

The ROM-NNFM repository (https://github.com/WybeSesink/ROM-NNFM) was the general workspace during the project. Analysis and tests were all made here. It also includes the implementation of adaptive sampling on the actual problem. It also includes the Emcee notebook used for Bayesian inference. Though the analyses show the method well, the code is likely not of much use for reuse.