International Journal of Computational Methods

Surrogate Modeling with Gaussian Processes for an Inverse Problem in Polymer Dynamics

Pankaj Chouhan and Sachin Shanbhag*

Department of Scientific Computing, Florida State University, Tallahassee, Florida 32304, United States of America

Received (Day Month Year)
Revised (Day Month Year)

When rheological models of polymer blends are used for inverse modeling, they can characterize polymer mixtures from rheological observations. This requires repeated evaluation of potentially expensive rheological models. We explored surrogate models based on Gaussian processes (GP-SM) as a cheaper alternative for describing the rheology of polydisperse binary blends. We used the time-dependent diffusion double reptation (TDD-DR) model as the true model; it takes a 5-dimensional input vector specifying the binary blend as input, and yields a function called the relaxation spectrum as output. We used the TDD-DR model to generate training data of different sizes $n = 30 - 1600$, via Latin hypercube sampling. The optimal values of the GP-SM hyper-parameters, assuming a separable covariance kernel, were obtained by maximum likelihood estimation. The GP-SM interpolates the training data by design, and offers reasonable predictions of relaxation spectra with uncertainty estimates. In general, the accuracy of GP-SMs improves as the size of the training data n increases, as does the cost for training and prediction. The optimal hyper-parameters were found to be relatively insensitive to n. Finally, we considered the inverse problem of inferring the structure of the polymer blend from a synthetic dataset generated using the true model. Surprisingly, the solution to the inverse problem obtained using GP-SMs and TDD-DR were qualitatively similar. GP-SMs can be several orders of magnitude cheaper than expensive rheological models, which provides a proof-of-concept validation for using GP-SMs for inverse problems in polymer rheology.

Keywords: surrogate model; Gaussian processes; separable kernel; inverse modeling.

1. Introduction

Synthetic polymers power the modern economy through their use in diverse sectors such as aerospace, medicine, automobile, electronics etc. Their properties can be tailored to specific applications not only by changing the chemistry of the monomers, but also by mixing or *blending* different polymers in a process that is analogous to alloying in metals. Another important reason for their popularity is the ease

* *sshanbhag@fsu.edu*

with which they can be processed. Typically, such processing is carried out in the molten state, which makes it imperative to develop an understanding of the underlying stress-strain relationships. Polymer rheology is the science that systematically studies these phenomena, and deals with the flow and deformation of these materials [Dealy and Larson (2006)].

Molecular models of polymer rheology take the molecular structure of a polymer mixture as input, and yield the linear or nonlinear rheology of the sample as output [Bird *et al.* (1987); Larson (1988)]. In the order of increasing spatial and temporal resolution, these models may be arranged from atomistic molecular dynamics and coarse-grained molecular dynamics on one end [Binder (1986); Kremer and Grest (1990); Kröger and Hess (2000); Likhtman *et al.* (2007)], through lattice models like the bond-fluctuation model and slip-link and slip-spring models [Carmesin and Kremer (1988); Schieber and Andreev (2014); Shaffer (1994); Subramanian and Shanbhag (2008); Tzoumanekas and Theodorou (2006)], to the mean-field tube model, which remains the most popular theory of polymer dynamics [Bird and Giacomin (2016); Larson *et al.* (2007); Masubuchi (2014)]. These different levels of modeling offer different trade-offs between accuracy and speed. By and large, as demands for accuracy increase, so do computational costs. Slip-link and slip-spring models offer a compelling practical balance accuracy and speed; nevertheless, the CPU time required to run a single simulation may vary between minutes to a day or more [Likhtman (2005); Masubuchi *et al.* (2008); Schieber and Andreev (2014); Shanbhag (2019a)].

Using predictive models of polymer rheology, processing methods can be rationally engineered to shape polymer melts into final products. Interestingly, due to the extreme sensitivity of rheology to molecular structure, these models can alternatively be used to characterize molecular weight distributions, long-chain branching, and the presence of contaminants [Bird and Giacomin (2016); Janzen and Colby (1999); Larson *et al.* (2007); Shanbhag (2012)]. This "inverse" use of molecular models is often called analytical rheology.

In analytical rheology, we measure the rheology of an unknown sample, and figure out the plausible polymer mixtures that might account for the measured rheology. This is a nonlinear inverse problem that requires repeated evaluations ($\sim 10^3 - 10^6$) of the rheological model [Shanbhag (2010, 2011, 2012); Takeh *et al.* (2011)]. In such situations, the need for fast approximations to the molecular models become imperative. These approximations are often called surrogate models, emulators, metamodels, or response surface models. In this work, we use the label *surrogate model* (SM) to refer to these fast approximations [Deng *et al.* (2020); Frangos *et al.* (2010); Goldstein and Rougier (2006); Kennedy and O'Hagan (2000, 2001); Li *et al.* (2016); Santner *et al.* (2018)].

1.1. *Gaussian Processes as Surrogate Models*

SMs are statistical models that seek to mimic the input-output relationship, $\boldsymbol{x} \rightarrow \boldsymbol{h}$, of computationally expensive (molecular) simulations. Loosely speaking, the input $\boldsymbol{x}$ represents the structure of the polymer mixture in our case, while the output $\boldsymbol{h}$ represents the rheology. Let m represent the molecular model, so that $\boldsymbol{h} = m(\boldsymbol{x})$. Ideally, the SM $\hat{m}$ approximates this model at a fraction of the computational cost. Since $\hat{m}$ is a statistical model it provides a point estimate $\hat{\boldsymbol{h}} = \mathbb{E}[\hat{m}(\boldsymbol{x})] \approx \boldsymbol{h}$, where $\mathbb{E}[.]$ denotes the expected value. We shall make these quantities more precise for our particular application in section 2.3.

The idea is then to use these cheaper SMs in lieu of the computationally more demanding simulations for tasks such as feasibility analysis for rational design optimization, global sensitivity analysis to identify the most important inputs, inverse modeling to obtain plausible inputs that might explain an observed output, etc. [Bhosekar and Ierapetritou (2018); Santner *et al.* (2018)]. The earliest SMs studied were univariate, $h = m(\boldsymbol{x})$ where $m : \mathbb{R}^d \rightarrow \mathbb{R}$. That is, the output of the simulation with potentially multiple inputs denoted by the vector $\boldsymbol{x} = [x^{(1)}, ..., x^{(d)}]$ is a scalar. However, interest in SMs for multiple outputs, $\boldsymbol{h} = m(\boldsymbol{x})$ where $m : \mathbb{R}^d \rightarrow \mathbb{R}^N$ has recently surged [Bayarri *et al.* (2007); Conti and O'Hagan (2010); Fricker *et al.* (2013); Higdon *et al.* (2008); Álvarez *et al.* (2012); McFarland *et al.* (2008); Tan (2018)]. We can classify these SMs into two categories: (i) *multiple-type* output, where the components of the output $\boldsymbol{h} = [h^{(1)}, \cdots, h^{(N)}]$ represent different types of quantities, and (ii) *field* outputs, where the components represent quantities over a continuous spatial or temporal field. The application considered here falls in the second category: the output represents a continuous function of time, and consecutive components of the output vector represent successive observations at discrete time intervals.

The use of Gaussian process (GP) regression is a popular surrogate modeling technique [Rasmussen and Williams (2006)]. It provides a systematic and flexible framework to emulate simulators with multivariate input and output variables. When used within a regression framework, training data is first accumulated by simulating the true model at n training points $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n$. We use superscripts $x^{(i)}$ to denote components of a vector $\boldsymbol{x}$, and subscripts $\boldsymbol{x}_i$ to denote different instances of $\boldsymbol{x}$, when there is a possibility of confusion. The hyper-parameters of the GP are typically inferred using maximum-likelihood estimation (MLE). At this point the SM is trained, and can be used to make predictions at a test input point, $\boldsymbol{x}_*$. See supporting information (SI) section **??** for an illustration of these steps for a standard univariate case.

GPs have two features that make them particularly attractive as SMs: (i) they interpolate the true model at the training points; thus $\boldsymbol{h}_i = m(\boldsymbol{x}_i) = \mathbb{E}[\hat{m}(\boldsymbol{x}_i)]$ for $i = 1, \cdots, n$ and (ii) they provide uncertainty estimates $\hat{\boldsymbol{\sigma}}_*^2 = \mathbb{V}[\hat{m}(\boldsymbol{x}_*)]$, where $\mathbb{V}[.]$ denotes the variance in the SM prediction at input point $\boldsymbol{x}_*$. Thus, the SM provides us with a prediction that takes the form $\hat{\boldsymbol{h}}_* \pm \hat{\boldsymbol{\sigma}}_*$, where $\hat{\boldsymbol{h}}_* = \mathbb{E}[\hat{m}(\boldsymbol{x}_*)]$ represents

the expected value or mean prediction of the SM, and $\hat{\boldsymbol{\sigma}}_*^2$ represents the expected variance. Thus, we can visualize GP-based SMs (abbreviated henceforth as GP-SM) as sophisticated interpolants with built-in uncertainty quantification.

1.2. *Motivation and Scope*

Typical methods of polymer synthesis result in polydispersity around the target chain length. Here, we explore a SM to predict the viscoelasticity of a blend of two polydisperse components. We label these materials as binary blends, and describe them in section 2.

Since the focus of this work is the development of a GP-SM and its use in analytical rheology, we use a somewhat simple theory called the time-dependent diffusion-double reptation (TDD-DR) theory as the first part of our simulation model. The simplicity allows us to easily generate large training datasets, and also test inverse modeling using the SM. In the future, it can be swapped with a more accurate and expensive model. TDD-DR is described in section 2.1; it takes in a description of a binary blend $\boldsymbol{x}$, and yields the linear viscoelasticity in terms of the normalized stress relaxation function $\phi(t)$. The stress relaxation function varies over several orders of magnitude and is not particularly smooth; it decays exponentially fast at long times. Therefore, a smoother function called the relaxation spectrum $h(s)$, which can be extracted from $\phi(t)$ using a program called pyReSpect [Shanbhag (2019b)], is used as the direct target for GP regression here (see figure 1a). The relationship between $\phi(t)$ and $h(s)$, and the computational methods used are described in sections 2.1 and 2.2.

Incorporating field outputs like $\phi(t)$ or $h(s)$ as the output of a GP presents its own challenges. Here, we discretize the spectrum on a fine mesh, using N grid points. The hyper-parameters of a GP-SM are learned from training samples $(\boldsymbol{x}_i, \boldsymbol{h}_i)$ with $i = 1, 2, \cdots, n$ that are generated from the true model (figure 1b). Naive GP regression involves inverting an $nN \times nN$ matrix during each iteration in the MLE estimation of GP hyper-parameters, which becomes overwhelming for typical values, $n \approx 100 - 1000$ and $N \approx 100$. Thus, we assume a *separable covariance kernel* as an approximation, to reduce the computational burden from $\mathcal{O}(n^3N^3)$ to $\mathcal{O}(n^3) + \mathcal{O}(N^3)$. This technique is described in section 2.3.

Once the optimal hyper-parameters $\hat{\boldsymbol{\theta}}$ are learned, the optimized GP-SM can be applied to predict the relaxation spectrum $\boldsymbol{h}_*$ at any test point $\boldsymbol{x}_*$ (figure 1c). Inverse modeling involves carrying out this step in reverse; i.e. instead of information about the polymer mixture $\boldsymbol{x}_*$, we are given an observed spectrum $\boldsymbol{h}_o$ and expected to describe the polymer mixtures that are approximately consistent with the observation (figure 1d). Note that the forward model $\boldsymbol{h} = m(\boldsymbol{x})$ is unique, but the inverse model $\boldsymbol{x} = m^{-1}(\boldsymbol{h})$ is not. If this were not the case, then the SM could be directly trained to mimic the inverse model. We evaluate the predictive abilities of the GP-SM in section 3.2, and demonstrate its use in inverse modeling with a single illustrative example in section 3.4.

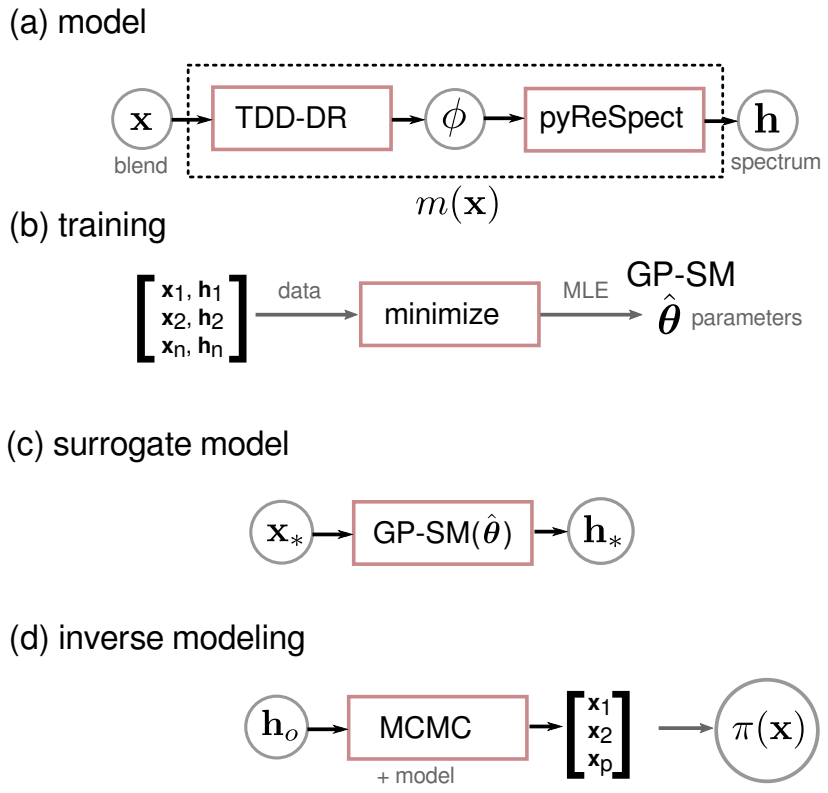
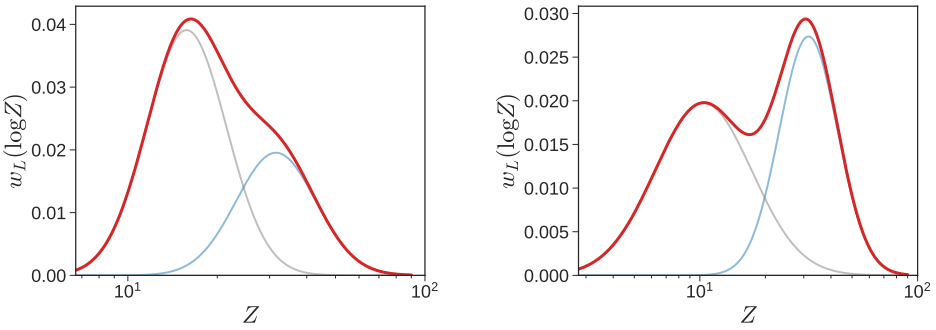


Fig. 1. Schematic diagram showing (a) the true model $m(\boldsymbol{x})$ (dotted line), which includes the TDD-DR model and the pyReSpect program. It takes in blend information in the form of $\boldsymbol{x} = [\bar{Z}_1, \bar{Z}_2, \rho_1, \rho_2, w_1]$ and yields the resulting relaxation spectrum; (b) training data generated using $m(\boldsymbol{x})$ is used to fit GP-SM hyper-parameters $\hat{\boldsymbol{\theta}}$ using maximum likelihood estimation; (c) the trained GP-SM model can be used in lieu of the true model at arbitrary input points $\boldsymbol{x}_*$; (d) in inverse modeling, the probability distribution of possible blends schematically denoted by $\pi(\boldsymbol{x})$ that are consistent with an observed spectrum $\boldsymbol{h}_o$ are sampled using Markov-chain Monte Carlo in conjunction with either the true or surrogate models.

We find that the optimal hyper-parameters $\hat{\boldsymbol{\theta}}$ are relatively insensitive to the size of the training data n. Nevertheless, the accuracy of GP-SMs improves as n increases, as does the cost for training and prediction. The solution to the inverse problem obtained using GP-SMs and TDD-DR are found to be qualitatively similar. Since GP-SMs are much cheaper than expensive rheological models, this study offers a proof-of-concept validation for using GP-SMs in analytical rheology.

2. Methods

Industrial synthesis of polymers leads to mixtures that contain polymer chains of different lengths. This distribution of chain lengths or molecular weights is called the molecular weight distribution (MWD). A popular parameterization used to describe



(a) $\rho_2 = 1.1$ and $w_1 = 0.5$ (b) $\rho_2 = 1.3$ and $w_1 = 0.7$

Fig. 2. MWD of a blend (thick red line) with $\bar{Z}_1 = 40$ and $\bar{Z}_2 = 20$. The polydispersity of the first component is held fixed at $\rho_1 = 1.1$. The blend MWD is resolved into contributions from the first (thin blue) and second (thin gray line) components via Eq. 3.

MWDs is the lognormal distribution which is given by,[cite L. H. Peebles, Molecular Weight Distributions in Polymers, Vol. 18, Interscience Publishers, New York 1971.]

$$\pi(\log Z; \mu_Z, \sigma_Z) = \frac{1}{\sqrt{2\pi\sigma_Z^2}} \exp\left(-\frac{(\log Z - \mu_Z)^2}{2\sigma_Z^2}\right). \tag{1}$$

This distribution in Eq. (1) is normalized so that $\int p(\log Z)\, d\log Z = 1$. The parameters of the lognormal distribution σ_Z^2 and μ_Z can be related to weight-averaged number of entanglements $\bar{Z}$ and the polydispersity ρ characterizing the MWD via,

$$\begin{aligned} \sigma_Z^2 &= \log \rho \\ \mu_Z &= \log \bar{Z} + \frac{1}{2} \log \rho. \end{aligned} \tag{2}$$

Thus, the overall MWD a binary blend with polydisperse components can be written as,

$$w_L(\log Z) = w_1\, \pi(\log Z; \mu_{Z,1}, \sigma_{Z,1}) + w_2\, \pi(\log Z; \mu_{Z,2}, \sigma_{Z,2}). \tag{3}$$

where $\mu_{Z,i}$ and $\sigma_{Z,i}$ are obtained from Eq. (2) with $\bar{Z} = \bar{Z}_i$ and $\rho = \rho_i$ for components $i = 1$ and 2. Thus, the input vector $\boldsymbol{x}$ can describe a wide array of mixtures of polymers. Two examples are illustrated in figure 2.

2.1. *Time-Dependent Diffusion-Double Reptation Model*

The three mechanisms of relaxation in polymers at equilibrium are reptation, contour-length fluctuations, and constraint release [de Gennes (1979); Doi and Edwards (1986)]. The double reptation theory models constraint release by stipulating that an entanglement between two chains relaxes as soon as one participant diffuses away, and that reptation time is unaffected by constraint release [de Cloizeaux (1988, 1990); Tsenoglou (1987, 1991)]. Despite its simplicity, it is useful in describing the

linear viscoelasticity of polydisperse systems [Mead (1996); Read *et al.* (2018); van Ruymbeke *et al.* (2007); Wasserman and Graessley (1992); Watanabe *et al.* (2004)]. In the double reptation theory, $\phi(t)$ is related to the MWD by,

$$\phi(t) = \frac{G(t)}{G_N^0} = \left(\int_0^\infty w_L(Z)\, F(t,Z)\, d\log Z \right)^\beta . \tag{4}$$

Where $G(t)$ is the shear relaxation modulus, G_N^0 is the plateau modulus, and β is the "double reptation" mixing exponent. Originally, $\beta = 2$ and $F(t,Z) = \exp(-t/\beta\tau_d)$, with $\tau_d = \tau_0 Z^3$ or $\tau_0 Z^{3.4}$ was commonly used [Tsenoglou (1991)]. However, for polydisperse high molecular weight polymers, the time-dependent diffusion double reptation (TDD-DR) model, [de Cloizeaux (1990)] with $\beta = 2.25$ was empirically found to be more accurate [van Ruymbeke *et al.* (2002)]. In TDD-DR, the kernel is replaced by,

$$F(t,Z) = \frac{8}{\pi^2} \sum_{p=\text{odd}} \frac{1}{p^2} \exp\left(-p^2 U(t,Z)\right), \tag{5}$$

where,

$$U(t,Z) = \frac{t}{\tau_d} + \frac{1}{H} g\left(\frac{Ht}{\tau_d}\right). \tag{6}$$

with $\tau_d = \tau_0 Z^3$, $H = Z/Z^*_{\text{TDD}}$, and [Shanbhag (2020a)],

$$g(x) = \sum_{n=1}^{\infty} \frac{1-\exp(-n^2 x)}{n^2} \approx \frac{\pi^2}{6} \operatorname{erf}(0.865\sqrt{x}). \tag{7}$$

In this work, we set the relaxation time of an entanglement strand $\tau_0 = 1$ sec, and $Z^*_{\text{TDD}} = 10$. With this choice, τ_0 is effectively the unit of time.

2.2. *Extraction of the Relaxation Spectrum*

The mathematical relationship between the normalized stress relaxation and the corresponding relaxation spectrum $h(s)$ is given by a Fredholm integral equation of the first kind,

$$\phi(t) = \int_{-\infty}^{\infty} h(s)\, e^{-t/s} d\log s. \tag{8}$$

The process of inferring $h(s)$ from $\phi(t)$ is non-trivial, but well-studied because it is a problem of central importance in linear viscoelasticity. Here, we use the publicly available python program pyReSpect, which implements non-linear Tikhonov regularization using a Bayesian criterion, to infer $h(s)$ [(Shanbhag, 2019b, 2020b; Takeh and Shanbhag, 2013)]. The regularization method used to extract the spectrum from stress relaxation measurements is described in detail in Shanbhag (2019b), and the implementation of the Bayesian criterion is fleshed out in Shanbhag (2020b). Here,

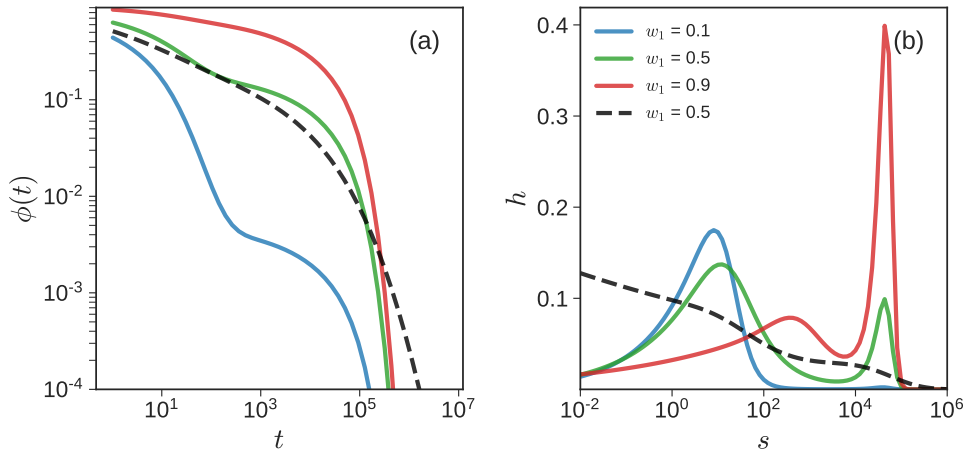


Fig. 3. Solid lines depict the (a) normalized stress relaxation functions $\phi(t)$ and (b) relaxation spectrum $h(s)$ of a bidisperse blend with chain lengths $\bar{Z}_1 = 50$ and $\bar{Z}_2 = 5$. The components are relatively monodisperse ($\rho_1 = \rho_2 = 1.01$). The weight fraction w_1 is varied between 0.1 and 0.9. The dashed lines shows how the response corresponding to $w_1 = 0.5$ changes when the individual components are polydisperse ($\rho_1 = \rho_2 = 1.5$).

we represent the spectrum on a fine grid using $N = 100$ (logarithmically) equispaced grid points between $s_{\min} = s^{(1)} = 10^{-2}$ and $s_{\max} = s^{(N)} = 10^{7}$,

$$\frac{s^{(i)}}{s_{\min}} = \left(\frac{s_{\max}}{s_{\min}}\right)^{\frac{i-1}{N-1}}. \tag{9}$$

We keep this grid constant and independent of the input $\boldsymbol{x}$. Thus, given an input $\boldsymbol{x}$, we use the TDD-DR model to obtain $\phi(t)$, and pyReSpect to extract the relaxation spectrum via Eq. (8). As shown in fig. 1a, the output of this pipeline is thus an N-dimensional vector $\boldsymbol{h} = [h^{(1)}, \cdots, h^{(N)}]$ defined on the constant grid, $\boldsymbol{s} = [s^{(1)}, \cdots, s^{(N)}]$, where $h^{(i)} = h(s^{(i)})$.

Figure 3 illustrates $\phi(t)$ and $h(s)$ for a bidisperse blend with $\bar{Z}_1 = 50$ and $\bar{Z}_2 = 5$. Solid lines represent a blend with relatively monodisperse components, $\rho_1 = \rho_2 = 1.01$. When $w_1 = 0.1$, we observe a two step relaxation in $\phi(t)$; the first step corresponds to the relaxation of the short chains, while the second step corresponds to the longer chains. As w_1 increases to 0.5 the height of the first step decreases while the terminal relaxation time increases due to the larger fraction of long chains. The spectrum corresponding to this blend shows two clear peaks corresponding to the two fractions, unlike the $w_1 = 0.1$ case, where the peak corresponding to $\bar{Z}_1$ is barely perceptible. As w_1 increases to 0.9, the contribution of $\bar{Z}_2$ to $\phi(t)$ is barely noticeable, while the peak attributable to $\bar{Z}_1$ becomes much stronger in $h(s)$. In all cases, the terminal response is marked by an exponentially fast decay in $\phi(t)$.

The dashed lines show the effect of increased polydispersity. When $\rho_1 = \rho_2 = 1.5$

with $w_1 = 0.5$, the MWDs of the individual components are no longer distinct, and show a significant overlapping region. Consequently, the "steps" in $\phi(t)$ and the "peaks" in $h(s)$ become diffuse. Due to the large polydispersity, a small fraction of the chains can get quite long and dominate the terminal response.

2.3. *Surrogate Modeling with Gaussian Processes*

Thus, the combination of TDD-DR and pyReSpect specifies the true simulation model. Given an input $\boldsymbol{x} = [\bar{Z}_1, \bar{Z}_2, \rho_1, \rho_2, w_1]$, it yields the relaxation spectrum on a fixed grid represented by the N-dimensional vector $\boldsymbol{s}$. We may represent this model as $\boldsymbol{h} = m(\boldsymbol{x})$, where $\boldsymbol{h}$ is an N-dimensional output vector. We can use this modeling pipeline to generate training data for n different input vectors $\boldsymbol{x}_i$ resulting in output training data $\boldsymbol{h}_i$, for $1 \leq i \leq n$. As shown schematically in figure 1b, training a GP involves determining the optimal value $\hat{\boldsymbol{\theta}}$ for the hyper-parameters.

In principle, the function describing relaxation spectra for binary blends $h(s; \boldsymbol{x})$ can be modeled with a GP by assuming the response to be a scalar function of a 6-dimensional input vector $\boldsymbol{z} = (s, \boldsymbol{x})$. However, this neglects the structure of the problem (the output grid is conserved), and incurs a very high computational cost $\mathcal{O}(N^3 n^3)$ during the optimization of GP parameters. To address both these issues, we use a separable covariance kernel where the contributions from the s and $\boldsymbol{x}$ co-ordinates are separately incorporated as,

$$k(\boldsymbol{z}, \boldsymbol{z}') = \sigma_h^2 R_s(s, s')\, R_x(\boldsymbol{x}, \boldsymbol{x}'), \tag{10}$$

where R_s and R_x are correlation functions, and σ_h^2 captures the variance. Due to the smoothness of the relaxation spectrum, we use a squared exponential correlation to define a $N \times N$ **grid correlation matrix** $\boldsymbol{R}_s$ whose (i, j) element is given by,

$$[\boldsymbol{R}_s]_{ij} = R_s(s^{(i)}, s^{(j)}) = \exp\left(-\frac{(\log s^{(i)} - \log s^{(j)})^2}{2\gamma_s^2}\right). \tag{11}$$

γ_s is a "length-scale" parameter associated with R_s. Since we are unsure about the smoothness properties with respect to input parameters, we use a Matérn correlation function with smoothness parameter set to 3/2 [Santner *et al.* (2018); Tan (2018)]. We build a $n \times n$ **input correlation matrix** $\boldsymbol{R}_x$, which is assumed to be a product of Matern kernels over the $d = 5$ dimensions of $\boldsymbol{x}$. If we denote the components using superscripts, $\boldsymbol{x}^{(1)} = \bar{Z}_1$, $\boldsymbol{x}^{(2)} = \bar{Z}_2$, $\boldsymbol{x}^{(3)} = \rho_1$, $\boldsymbol{x}^{(4)} = \rho_2$, and $\boldsymbol{x}^{(5)} = w_1$, then the (i, j) element is given by

$$[\boldsymbol{R}_x]_{ij} = R_x(\boldsymbol{x}_i, \boldsymbol{x}_j) = \prod_{k=1}^{5} (1 + p_k)\, e^{-p_k}, \text{ with } p_k = \frac{\sqrt{6}}{\gamma_{x,k}} \left| x_i^{(k)} - x_j^{(k)} \right| \tag{12}$$

where $\gamma_{x,1}, \gamma_{x,2}, \cdots, \gamma_{x,5}$ correspond to the length-scale parameters associated with each component of the input vector $\boldsymbol{x}$. Let $\boldsymbol{\theta} = (\sigma_h^2, \gamma_s, \gamma_{x,1}, \cdots, \gamma_{x,5})$ denote the 7 unknown kernel parameters or GP hyper-parameters, which are to be jointly estimated from the training data.

2.3.1. *Estimation of Kernel Parameters*

We stack the output of the training data into an $Nn \times 1$ column vector $\boldsymbol{H} = [\boldsymbol{h}_1, \cdots, \boldsymbol{h}_n]^T$. We standardize the output data-vector, $\boldsymbol{H} \leftarrow \boldsymbol{H} - \bar{H}$, by subtracting the mean value $\bar{H} = \sum_{i,j} h_i^{(j)}/(Nn)$. We implicitly define the $Nn \times Nn$ covariance matrix $\boldsymbol{K} = \sigma_h^2 \cdot \boldsymbol{R}_x \otimes \boldsymbol{R}_s$, where $\otimes$ denotes a Kronecker product.

Assuming a zero-mean GP, the log-likelihood $\mathcal{L}(\boldsymbol{\theta})$ of observing the training data $\boldsymbol{H}$ with kernel parameters $\boldsymbol{\theta}$ can formally be shown to be [Rasmussen and Williams (2006)],

$$\mathcal{L}(\boldsymbol{\theta}) = -\frac{1}{2}\boldsymbol{H}^T\boldsymbol{K}^{-1}\boldsymbol{H} - \frac{1}{2}\log|\boldsymbol{K}| - \frac{Nn}{2}\log 2\pi. \tag{13}$$

The likelihood depends on the kernel parameters implicitly through the covariance matrix $\boldsymbol{K}(\boldsymbol{\theta})$. For a given training set with fixed N and n, the last term can be ignored in the optimization. The first and second terms can be identified with *data-fit* $\text{DF} = \boldsymbol{H}^T\boldsymbol{K}^{-1}\boldsymbol{H}$, and *complexity penalty* $\text{CP} = \log|\boldsymbol{K}|$, respectively. The cost function $\mathcal{C}$ can be defined as the sum of these two contributions.

$$\mathcal{C}(\boldsymbol{\theta}) = \text{DF} + \text{CP} = \boldsymbol{H}^T\boldsymbol{K}^{-1}\boldsymbol{H} + \log|\boldsymbol{K}|\,. \tag{14}$$

The kernel parameters $\hat{\boldsymbol{\theta}}$ that maximize $\mathcal{L}$ (or minimize $\mathcal{C}$) by optimally trading off data-fit and complexity penalty give us the maximum likelihood estimates (MLE). In practice, we use the truncated Newton algorithm (TNC in scipy.optimize) with 10 well-dispersed initial conditions to mitigate entrapment in local minima. We require all parameters to be non-negative, and choose $\hat{\boldsymbol{\theta}}$ corresponding to the best (maximum $\mathcal{L}$) of these 10 estimates.

Computationally, separability of the covariance makes the problem tractable. Otherwise, for typical values of $n \approx 10^2 - 10^3$, and $N \approx 100$, computing $\boldsymbol{K}^{-1}$ would be a formidable $\mathcal{O}(N^3n^3)$ operation at each iteration of the optimizer. Fortunately, the properties of a Kronecker product allow us to compute the inverse and log determinant of the covariance matrix in terms of the inverse and covariance of the smaller correlation matrices,

$$\begin{aligned} \boldsymbol{K}^{-1} &= (\boldsymbol{R}_x^{-1} \otimes \boldsymbol{R}_s^{-1})/\sigma_h^2 \\ \log|\boldsymbol{K}| &= N\log|\boldsymbol{R}_x| + n\log|\boldsymbol{R}_s| + Nn\log\sigma_h^2. \end{aligned} \tag{15}$$

This reduces the computational burden to $\mathcal{O}(N^3 + n^3)$ required for separately inverting the correlation matrices.

2.3.2. *Prediction*

Once the kernel parameters $\hat{\boldsymbol{\theta}}$ are regressed and the GP describing the posterior distribution is calibrated, it can be used to predict the response at a test input point $\boldsymbol{x}_*$. A point-estimate can be obtained from the expected value at the test point [Rasmussen and Williams (2006)],

$$\hat{\boldsymbol{h}}_* = \hat{\boldsymbol{K}}_*^T\hat{\boldsymbol{K}}^{-1}\boldsymbol{H}, \tag{16}$$

where $\hat{\boldsymbol{K}}_* = \sigma_h^2(\hat{\boldsymbol{r}}_x^* \otimes \hat{\boldsymbol{R}}_s)$ and $\hat{\boldsymbol{r}}_x^*$ is the $n \times 1$ column vector, whose i^{th} element is $[\hat{\boldsymbol{r}}_x^*]_i = R_x(\boldsymbol{x}_i, \boldsymbol{x}_*)$. Quantities decorated with a "hat" use the MLE parameters $\hat{\boldsymbol{\theta}}$; for example, $\hat{\boldsymbol{K}}$ denotes the covariance matrix using these parameters. The variance vector associated with this mean prediction is given by,

$$\hat{\boldsymbol{\sigma}}_*^2 = \sigma_h^2 \mathbf{1} - \text{diag}\left(\hat{\boldsymbol{K}}_*^T \hat{\boldsymbol{K}}^{-1} \hat{\boldsymbol{K}}_*\right), \tag{17}$$

where $\mathbf{1}$ is an $N \times 1$ column vector with all elements equal to 1, and diag$(\cdot)$ extracts the diagonal vector from an $N \times N$ square matrix.

Note that the separable covariance structure assumed for the kernel is somewhat restrictive. An alternative is to use a GP based on a Karhunen-Loève expansion [Azzi *et al.* (2019); Huang *et al.* (2001); Jain (1976); Phoon *et al.* (2005); Tan (2018, 2019)]. This method can use the eigendecomposition of the empirical covariance function obtained from training data, which makes it possible to describe non-stationary kernels naturally. Other advantages of this method includes computational efficiency, less restrictive assumptions, and more fine-grained uncertainty quantification. Preliminary work on our problem using this method, while promising, seems to yield results comparable with the separable covariance assumptions. However, more work is needed to make stronger claims and is currently underway.

3. Results

The first step in implementing GP-SM is selecting training data. In our work, we assume that the chain lengths are confined to $5 \leq \bar{Z}_2 < \bar{Z}_1 \leq 50$, and that the polydispersities are confined to $1.01 \leq \rho_1, \rho_2 \leq 1.5$. These limits are arbitrary, and may be changed as desired. The weight fraction w_1 is necessarily confined to $0 \leq w_1 \leq 1$. Note that we require $\bar{Z}_1 > \bar{Z}_2$ to uniquely define a binary blend with nonzero w_1; otherwise the positions of the two components can be interchanged.

We draw n training data $\boldsymbol{x} = (\bar{Z}_1, \bar{Z}_2, \rho_1, \rho_2, w_1)$ using Latin hypercube sampling [McKay *et al.* (1979)]. Since each training data point is 5-dimensional, it is not trivial to visualize. Figure 4 depicts projections of $n = 53$ samples on 3 different 2D planes (out of the possible ${}^5C_2 = 10$). A triangular pattern is observed in figure 4a because of the constraint $\bar{Z}_1 > \bar{Z}_2$. This is reflected in figure 4b, where the cloud of points appears to be concentrated in right half of the plot, but uniformly dispersed along the vertical (w_1) axis. Finally, in figure 4c, the points appear to be distributed uniformly.

We also standardize and internally rescale the input data $\boldsymbol{x}$ so that the upper limit of each component is of order 1. This involves dividing $\bar{Z}_1$ and $\bar{Z}_2$ by the maximum allowed value of 50. The other components of $\boldsymbol{x}$ are inherently $\sim \mathcal{O}(1)$. This normalization assists with the MLE estimation. We use this shifted data to infer the GP hyper-parameters, which are summarized in table 1 for the $n = 53$ training points shown in figure 4.

The model is trained in a few seconds, showing the computational advantage of using a separable covariance structure. We initialize 10 independent optimization

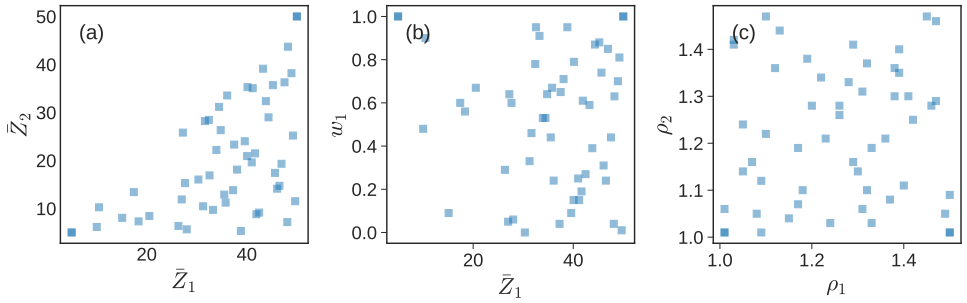


Fig. 4. Training data ($n = 53$) drawn using Latin hypercube sampling is projected on three different 2D planes.

Table 1. MLE estimates $\hat{\boldsymbol{\theta}}$ for $n = 53$

$\mathcal{C}(\hat{\boldsymbol{\theta}})$	-6.0×10^4	$\hat{\gamma}_{x,1}$	1.9
time (s)	3.4	$\hat{\gamma}_{x,2}$	0.62
$\hat{\sigma}_h^2$	3.1×10^{-3}	$\hat{\gamma}_{x,3}$	0.99
$\hat{\gamma}_s$	3.6×10^{-1}	$\hat{\gamma}_{x,4}$	5.0
		$\hat{\gamma}_{x,5}$	3.5

runs by randomly selecting the kernel parameters to the range [1, 3], and selecting the set that that leads to the largest $\mathcal{L}(\boldsymbol{\theta})$. We use multiple runs to avoid being trapped in local minima, and to overcome bad initial guesses in which the algorithm fails to converge. Figure **??** in supplementary information visualizes the negotiation between data-fit and complexity penalty and the convergence to a minimum.

3.1. *Interpolation and Prediction*

Figure 5 shows the output of the trained GP-SM for test datapoints. In figure 5a, $\boldsymbol{x}_* = (31.27, 10.45, 1.19, 1.38, 0.33)$ is chosen to be identical to one of the training data points. Unsurprisingly, the "estimated" spectrum $\hat{\boldsymbol{h}}_*$ overlaps with the true spectrum, due to the interpolation property of GPs. Furthermore, the uncertainty associated with the estimated spectrum is correctly identified as zero.

Figure 5b shows the prediction of the GP-SM at a unseen test data point $\boldsymbol{x}_*$ = (43.0, 18, 1.15, 1.34, 0.38). Unlike figure 5a, the true and estimated spectra no longer overlap perfectly. The shaded region around the mean estimate depicts the $\pm 2.5\hat{\boldsymbol{\sigma}}_*$ uncertainty band. In light of the uncertainty expressed by the prediction, the agreement between the two spectra appears to be reasonable.

Finally, figure 5c shows the $\phi(t)$ corresponding to these two spectra predictions obtained by numerical integration of Eq. (8). The mean prediction seems to be reasonably good until about $t \approx 10^5$. However the tiny bump seen in spectrum near $s \approx 10^4$ translates to significantly slower terminal relaxation for $\phi(t > 10^5)$.

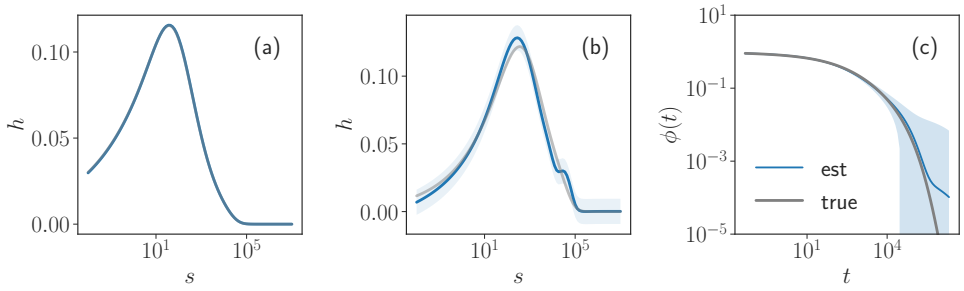


Fig. 5. (a) GP-SM prediction when $\boldsymbol{x}_* = (31.27, 10.45, 1.19, 1.38, 0.33)$ is selected as one of the training point coincides with the true model prediction. (b) This is not true for an unseen test point $\boldsymbol{x}_* = (43.0, 18, 1.15, 1.34, 0.38)$. The $\pm 2.5\hat{\boldsymbol{\sigma}}_*$ uncertainty interval is shown by the shaded region. The gray line is the output of the true model. (c) The $\phi(t)$ generated using the relaxation spectra predicted by the GP-SM from (b) is compared with the true $\phi(t)$.

The uncertainty bands for $\phi(t)$ are obtained from 1000 independent samples of $\mathbf{h}$ drawn from the trained GP. The bands are rather large and sufficiently reflect the deviation of the mean estimate from the true relaxation modulus.

3.2. *Evaluation of the GP-SM*

In order to quantitatively test the SM model with $\hat{\boldsymbol{\theta}}$ regressed from the $n = 53$ training data set, we generated $n_{\text{test}} = 251$ independent test samples for $\boldsymbol{x}_*$. These samples were also chosen using Latin hypercubic sampling, but with different random seeds. For the predictions, we used $\hat{\boldsymbol{\theta}}$ shown in table 1 along with Eq. (16) and Eq. (17). To quantitatively summarize our findings we define three evaluation metrics; the root mean squared error (RMSE), the median absolute deviation (MAD), and the coverage metric D_α.

For a particular test input $\boldsymbol{x}$, let the N-element vector $\Delta\boldsymbol{h} = \hat{\boldsymbol{h}}_* - \boldsymbol{h}^{\text{true}}$ captures the deviation between the predicted point-estimate ($\hat{\boldsymbol{h}}_*$) and true ($\boldsymbol{h}^{\text{true}}$) spectra. Then, the RMSE for a particular input is defined via the 2-norm of $\Delta\boldsymbol{h}$,

$$\text{RMSE} = \frac{1}{\sqrt{N}} \, ||\Delta\boldsymbol{h}| \, |_2. \tag{18}$$

If the absolute value of the i^{th} component of $\Delta\boldsymbol{h}$ is denoted by $|\Delta\boldsymbol{h}^{(i)}|$, then the MAD for a particular prediction is defined as,

$$\text{MAD} = \text{median} \left\{ |\Delta\boldsymbol{h}^{(1)}|, \cdots, |\Delta\boldsymbol{h}^{(N)}| \right\}. \tag{19}$$

Relative to the RMSE, we expect MAD to under-emphasize outliers, and estimate the *typical* deviation at a randomly selected grid point. These two metrics compare the deviation of the *mean* prediction or expected value from the true spectra.

The third metric D_α seeks to evaluate whether the uncertainty $\hat{\boldsymbol{\sigma}}_*$ associated with the mean prediction is appropriately calibrated. It is defined as the fraction

of $100\alpha\%$ credible intervals that contain the true value. A SM that is perfectly calibrated, and whose error follows a normal distribution, will have $D_\alpha \approx \alpha$, where $\alpha \in [0, 1]$, so that a plot of D_α v/s α is a diagonal straight line. Deviations from this trend quantitatively summarize the dispersion about the mean prediction, and helps us qualitatively asses whether the SM is over- or under-confident.

To compute D_α, we create a list of Nn_{test} elements, $\boldsymbol{\xi} = \{\Delta \boldsymbol{h}_j^{(i)} / \hat{\boldsymbol{\sigma}}_{*,j}^{(i)}\}$, where $1 \le i \le N$ and $1 \le j \le n_{\text{test}}$. For a perfectly calibrated model, these scaled deviations are expected to follow a unit normal distribution, which has a cumulative distribution function given by $\text{cdf}(z) = 0.5[1 + \text{erf}(z/\sqrt{2})]$. For a particular value of α, D_α is then the fraction of elements of $\boldsymbol{\xi}$ that are less than or equal to $\xi_\alpha = \sqrt{2}\,\text{erf}^{-1}(2\alpha - 1)$.

The mean and standard error for MAD and RMSE are estimated to be $4.5 \pm 0.5 \times 10^{-3}$ and $1.0 \pm 0.1 \times 10^{-2}$ respectively. To contextualize these numbers, the magnitude of $\boldsymbol{h}$ is $\mathcal{O}(0.1)$, so the relative error bars implied by MAD and RMSE are on average about 10 - 20 times smaller than the magnitude of $\boldsymbol{h}$.

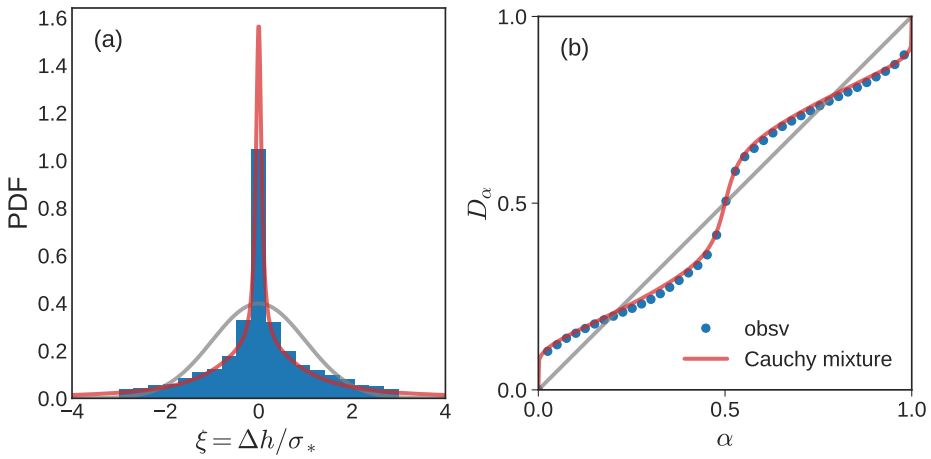


Fig. 6. (a) The empirical distribution of the scaled test deviations ξ (blue histogram) appears to follow a mixture of two Cauchy distributions (red line), rather than a normal distribution (gray line). (b) The deviation of D_α (blue symbols) from the straight line corresponding to a normal distribution reveals finer structure of the empirical data. The red line shows the theoretical D_α for the mixture of Cauchy distributions fitted to the empirical D_α.

Figure 6a shows the empirical distribution of the $n_{\text{test}}N$ elements of $\boldsymbol{\xi}$, representing scaled deviations. Unlike MAD or RMSE, which treat positive and negative deviations alike, the elements of $\boldsymbol{\xi}$ are *signed*. The observed histogram is symmetric about zero, indicating that positive and negative deviations about $\boldsymbol{h}^{\text{true}}$ are equally likely. Furthermore, $\boldsymbol{\xi}$ is also *scaled* by the estimated error $\hat{\boldsymbol{\sigma}}_*$. We might expect that these samples to follow a normal distribution with unit variance, depicted by

the gray line. However, relative to a normal distribution, the empirical distribution has more probability mass, *both* near the mean $\xi = 0$ and far away from the mean $\xi \to \pm\infty$.

This implies that for a large fraction of the samples, the observed error is actually smaller than that implied by a normal distribution. Simultaneously, the fraction of outliers is also significantly larger suggesting that the distribution is fat-tailed. At first glance, the empirical distribution appears to follow a Cauchy distribution whose PDF is given by,

$$\pi(x; \gamma_C) = \frac{1}{\pi\gamma_{\rm C}(1 + (x/\gamma_{\rm C})^2)}, \tag{20}$$

where the parameter γ_C controls the width of the distribution.

Finally, figure 6b shows D_α versus α. If the data were normally distributed, we would expect it to track the diagonal gray line. The observed D_α shown by the blue symbols hovers around the diagonal, intersecting it at $\alpha \approx 0.23$, 0.5, and 0.77, besides the two end-points. The intersection at $\alpha = 0.5$ reflects the symmetry of the distribution. For small values of $\alpha < 0.23$, D_α lies above the diagonal indicating the presence of a fat left tail. Far away from extreme negative outliers, the observed distribution has a dearth of samples relative to the standard normal. This causes D_α to "catch up" with the diagonal at $\alpha \approx 0.23$. This trend continues, and D_α goes below the diagonal between $0.23 \lesssim \alpha \lesssim 0.5$ as most of the probability mass is concentrated near the center which corresponds to $\alpha = 0.5$. This part is clearly visible in figure 6a as well.

The symmetry of the PDF about its center is reflected in the symmetry of D_α. For α in the range [0.50, 0.77], D_α lies above the diagonal, again reflecting the concentration of the probability mass near the center. For $\alpha \gtrsim 0.77$, D_α again slips below the diagonal reflecting the presence of positive outliers which eventually, and abruptly, pull up D_α to 1.

Recall that for a given α, the cutoff $\xi_\alpha = \sqrt{2}\,\mathrm{erf}^{-1}(2\alpha - 1)$. Thus, the theoretical D_α for a Cauchy distribution is given by the cumulative distribution function,

$$D_\alpha^{\rm Cauchy}(\alpha, \gamma_C) = \frac{1}{2} + \frac{1}{\pi}\tan^{-1}\left(\frac{\sqrt{2}\,\mathrm{erf}^{-1}(2\alpha - 1)}{\gamma_C}\right). \tag{21}$$

Unfortunately, a Cauchy distribution with a single γ_C parameter is not able to describe the observed D_α. It can either capture the center of the distribution near $\alpha = 0.5$, or the outliers near $\alpha = 0$ and 1. In fact, the true empirical distribution is best described by a weighted average of two Cauchy distributions which is shown by the red line in figure 6b,

$$D_\alpha \approx 0.3\, D_\alpha^{\rm Cauchy}(\alpha, 0.07) + 0.7\, D_\alpha^{\rm Cauchy}(\alpha, 1.03). \tag{22}$$

The corresponding PDF of the mixture of Cauchy distributions is also shown in figure 6a.

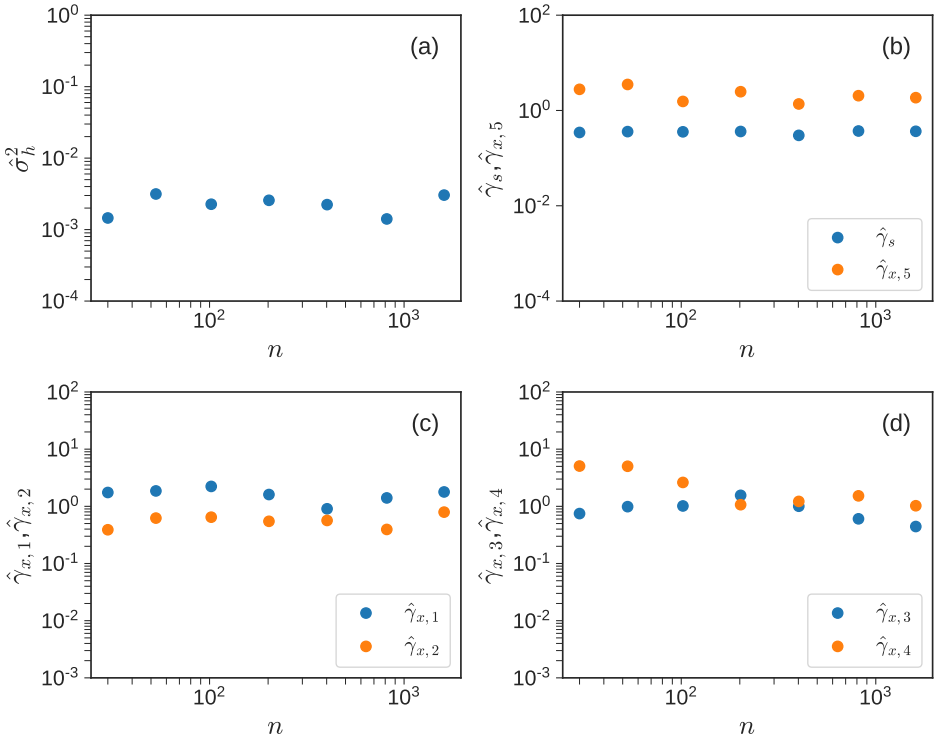


Fig. 7. Variation of MLE estimates $\hat{\boldsymbol{\theta}}$ with training data size n. Components of $\hat{\boldsymbol{\theta}}$ with comparable magnitudes are grouped together.

3.3. *Dependence on Size of Training Data*

Thus far, we used a rather small training data set with $n = 53$ samples. Here, we explore how GP hyper-parameters, performance metrics and computational cost change as the size of the training data is varied between $n = 30 - 1600$. The particular values of n chosen are 30, 53, 102, 202, 402, 815, and 1606.

3.3.1. *MLE Parameters*

Figure 7 shows the variation of MLE estimates $\hat{\boldsymbol{\theta}}$ corresponding to the 7 hyper-parameters as n is increased. Quantities with comparable magnitudes are plotted together. By and large, values of the components of $\hat{\boldsymbol{\theta}}$ appear to be relatively stable, and insensitive to n. This is fortunate because computationally, $\hat{\boldsymbol{\theta}}$ is easier to estimate from smaller training datasets. The relative stability of the parameters means that good initial guesses for $\hat{\boldsymbol{\theta}}$ of large datasets can be obtained by first solving the parameter estimation problem on a smaller representative subset (say $n \approx 100$) of the training data. Supplying a good initial guess can thus reduce the number of

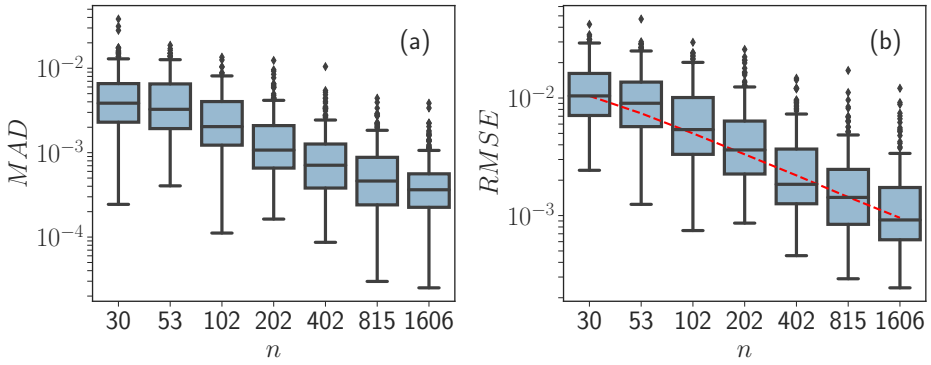


Fig. 8. Variation of (a) MAD, and (b) RMSE with training sample size n represented using box-plots. The red dashed line indicate the RMSE varies as $n^{-0.6}$. The number of test data is $n_{\text{test}} = 251$.

iterations required to estimate $\hat{\boldsymbol{\theta}}$ for large datasets, even as the cost per iteration increases.

The relative stability of the $\hat{\boldsymbol{\theta}}$ with n also helps with interpretation, since it captures general trends in the data. Meaning can be ascribed to the variance ($\hat{\sigma}_h^2 \approx \mathcal{O}(10^{-3})$) and length-scales ($\gamma$) associated with the individual variables. Thus, for example, $\gamma_s \approx \mathcal{O}(10^{-1})$ corresponds to extent of $\log s$ over which the spectrum decorrelates. This is quite different from $\gamma_{x,1}$ and $\gamma_{x,2} \approx \mathcal{O}(1)$ which correspond to the normalized chain lengths $\bar{Z}_1/50$ and $\bar{Z}_2/50$. The lengthscales corresponding to the polydispersities are $\gamma_{x,3}$ and $\gamma_{x,4} \approx \mathcal{O}(5)$ are comparable with that associated with w_1.

Remarkably, the values of $\hat{\boldsymbol{\theta}}$ regressed from the largest training data sets are consistent with those shown in figure 7 for $n \approx 10^2 - 10^3$. For context, training the $n = 53$ dataset costs 3.4 sec, compared with ≈ 3 days required for training the $n = 12480$ dataset (see section **??** in SI).

3.3.2. *Performance Metrics*

Figure 8 summarizes the variation of MAD and RMSE with the number n of training data points. At a particular value of n, a GP is trained to obtain the MLE parameters $\hat{\boldsymbol{\theta}}$ shown in figure 7. Let us denote a GP trained with n data points as GP-SM-n (e.g GP-SM-53 or GP-SM-1606).

We test the predictive ability of each trained GP-SM on $n_{\text{test}} = 251$ unseen test data points $\boldsymbol{x}_*$. At each individual test point, we obtain a MAD and RMSE using Eq. 19 and Eq. 18. The distribution of the MAD and RMSE over all the n_{test} test data is presented in the form of box-plots in figure 8.

As expected, the median values of both MAD and RMSE decrease with n. This indicates that the predictive ability of the GP-SM improves as it is trained on more

data. RMSE decreases with n, approximately as $n^{-0.6}$. The number of outliers corresponding to large values of MAD and RMSE remains stubbornly high, regardless of n. However, recall that both these metrics consider the discrepancy between the true spectrum and the point-estimate $\hat{\boldsymbol{h}}_*$, without regard to the expressed uncertainty.

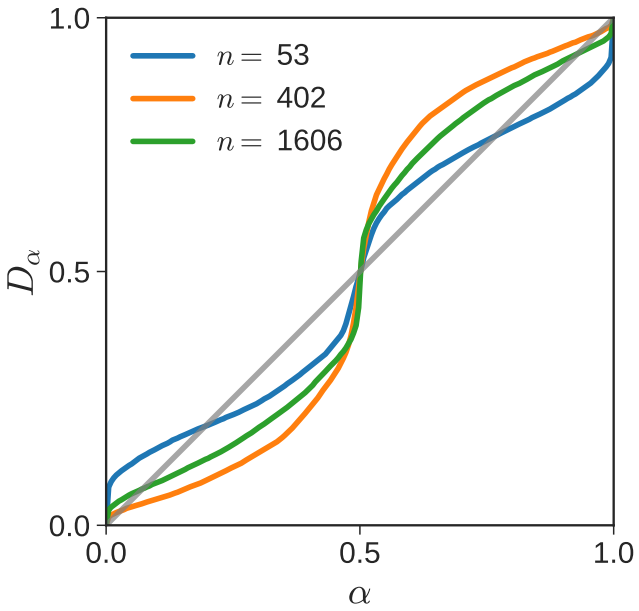


Fig. 9. This figure illustrates the deviation of D_α for $n = \{53, 402, 1606\}$. The gray line represents the D_α for a unit normal distribution. For $0.0 \leq \alpha \lesssim 0.5$, D_α for GP-SM-402 and GP-SM-1606 lie below the D_α for GP-SM-53, and for $0.5 \lesssim \alpha \leq 1$ they lie above the D_α for GP-SM-53. Therefore, the distributions of ξ for GP-SM-402 and GP-SM-1606 are more strongly peaked than that for GP-SM-53. This indicates that as n increases, the fraction of outliers decreases. The intersection of D_α curves with gray line at $\alpha \sim 0.5$ reflects the symmetry of the distributions.

Figure 9 compares the D_α at three different values of n. The blue line corresponding to GP-SM-53 was previously shown in Fig. 6b. The overall shape of the D_α curves is similar. The number of extreme outliers, observed as a "jump" near $\alpha = 0$ and 1 for GP-SM-53, decreases for GP-SM-402 and GP-SM-1606. Interestingly, the jump is slightly less pronounced for $n = 402$ compared to $n = 1606$. Currently, we do not have a good explanation for the lack of any trend once $n \gtrsim \mathcal{O}(100)$.

In general, the accuracy of GP-SMs is improved by using larger training datasets. A major impediment is that the computational cost increases rapidly with the size of the training data. Numerous workarounds have been devised to circumvent some of these issues, which can be adapted for our problem in the future. These in-

clude sparse [Furrer *et al.* (2006); Gneiting (2002); Kaufman *et al.* (2008)], low-rank [Hensman *et al.* (2013); Seeger *et al.* (2003); Snelson and Ghahramani (2006); Titsias (2009)], or local [Gramacy and Apley (2015); Nguyen-Tuong *et al.* (2008); Park and Apley (2018); Rasmussen and Ghahramani (2001); Snelson and Ghahramani (2007); Vanhatalo and Vehtari (2008)] approximations.

3.4. *Demonstration of SM for Inverse Modeling*

Once we have a "forward model" $\boldsymbol{h} = m(\boldsymbol{x})$ or an $\boldsymbol{h} \approx \hat{m}(\boldsymbol{x})$, we can consider using it for inverse modeling. Here, the original ($m(\boldsymbol{x})$) and approximate ($\hat{m}(\boldsymbol{x})$) models correspond to the TDD-DR and GP-SM, respectively, as the first step (see figure 1) of the forward model. One feature of analytical rheology that makes it very attractive for inverse modeling is that rheology is extremely sensitive to changes in molecular structure. This makes the inverse problem correspondingly robust [Shanbhag (2010)].

In the typical inverse modeling setup, we are provided an observation $\boldsymbol{h}_o$, and the goal is to make inferences about $\boldsymbol{x}$ by somehow inverting the forward model, viz. $\boldsymbol{x} = m^{-1}(\boldsymbol{h}_o)$. Nonlinear inverse problems often have multiple solutions. Thus, it is unwise to pose the inverse problem as an optimization problem, where we search for the input $\boldsymbol{x}$ to the forward model that results in the best-fit with the observation $\boldsymbol{h}_o$. A better strategy is to pose the question as a Bayesian inference problem. This avoids overfitting that may arise from potential uncertainties in the observed data or model specification.

For example, let $\boldsymbol{h}_o = m(\boldsymbol{x}_o)$ be the true observed spectrum resulting from a binary blend described by the input vector $\boldsymbol{x}_o$. Let us subsequently pretend that we do not know $\boldsymbol{x}_o$, and seek to find the distribution of binary blends that yield a spectrum consistent with $\boldsymbol{h}_o$. According to Bayes theorem, the distribution of $\boldsymbol{x}$ that is consistent with the observation $\boldsymbol{h}_o$, is given by

$$\pi(\boldsymbol{x}|\boldsymbol{h}_o) \propto \pi(\boldsymbol{h}_o|\boldsymbol{x})\pi(\boldsymbol{x}), \tag{23}$$

where the posterior distribution $\pi(\boldsymbol{x}|\boldsymbol{h}_o)$ is proportional to the product of the likelihood $\pi(\boldsymbol{h}_o|\boldsymbol{x})$ of observing $\boldsymbol{h}_o$ given $\boldsymbol{x}$, and the prior $\pi(\boldsymbol{x})$. We assume that the prior is uniform on the domain used to train GP-SMs, i.e. $\pi(\boldsymbol{x}) = \prod \pi_i(\boldsymbol{x}^{(i)})$. Thus, $\pi_3(x^{(3)} = \rho_1)$ and $\pi_4(x^{(4)} = \rho_2)$ are uniform distributions $U[1.01, 1.5]$. Similarly, $\pi_5(x^{(5)} = w_1) \sim U[0, 1]$ and $\pi_1(\bar{Z}_1) = U[5, 50]$. Since, $\bar{Z}_2 < \bar{Z}_1$, the conditional distribution $\pi_2(\bar{Z}_2|\bar{Z}_1) = U[5, \bar{Z}_1]$.

As an illustrative example, let us reconsider the test data point $\boldsymbol{x}_o = (43.0, 18, 1.15, 1.34, 0.38)$ originally encountered in figure 5. Thus, we take $\boldsymbol{h}_o$ (the spectrum shown by the gray line in figure 5b) as the "synthetic" observed data for this illustration. The goal is to infer the distribution of binary blends $\boldsymbol{x}$ that lead to the same or similar spectra.

First, we consider the inverse problem using the TDD-DR model. For many nonlinear models, inversion of synthetic output data does not lead uniquely to inputs

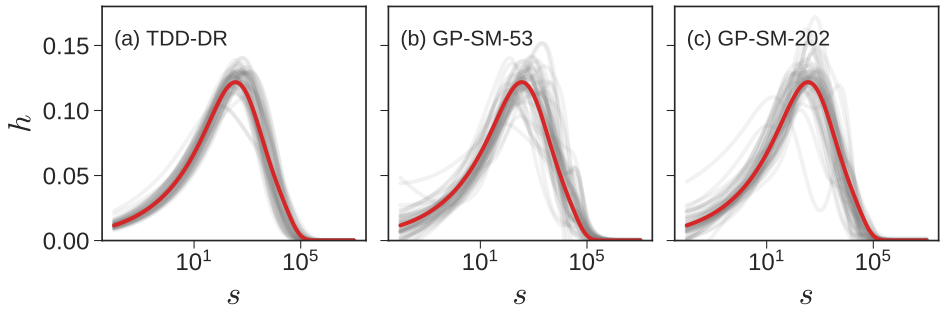


Fig. 10. Gray lines depict 50 of the 500 samples generated using the MCMC method for the (a) TDD-DR, (b) GP-SM based on $n = 53$ training data, and (c) GP-SM based on $n = 202$ training data. The solid line denotes the target $\boldsymbol{h}_o$ spectrum generated using $\boldsymbol{x}_o = (43.0, 18, 1.15, 1.34, 0.38)$.

used to generate that data. This will become apparent shortly. Suppose we guess an input $\boldsymbol{x}$, and use the forward model to compute the spectrum $\boldsymbol{h} = m(\boldsymbol{x})$. We can define the "distance" between $\boldsymbol{h}$ and the target spectrum $\boldsymbol{h}_o$ as the Euclidean distance (2-norm) between the two spectra,

$$d(\boldsymbol{x}) = \left\| \frac{\boldsymbol{h} - \boldsymbol{h}_o}{\boldsymbol{\sigma}} \right\|_2 , \tag{24}$$

where $\boldsymbol{\sigma}$ is the uncertainty associated with the model prediction. Using this distance function, we propose a simple form for the likelihood function,

$$\pi(\boldsymbol{h}_0|\boldsymbol{x}) \propto \exp\left(-\alpha_d d(\boldsymbol{x})\right), \tag{25}$$

where α_d is a scaling constant. Thus, evaluating the likelihood at a sample point $\boldsymbol{x}$ requires us to evaluate the forward model at that point, and compute the distance $d(\boldsymbol{x})$. By construction, the likelihood function is large when the distance between the two spectra is small. The form assumed here for $\pi(\boldsymbol{h}_0|\boldsymbol{x})$ is not unique, and any other reasonable form may be used instead.

When we use TDD-DR as the forward model, we take $\boldsymbol{\sigma}$ = constant, which has the effect of subsuming the uncertainty parameter into the constant α_d. To determine α_d, which calibrates the distance function, we perform a short precalibration run by randomly selecting 100 well-dispersed input points $\boldsymbol{x}$, and computing the distance function at each $\boldsymbol{x}$. We set $\alpha_d = 5/s_d$, where s_d is the standard deviation of the distances explored in the precalibration run.

We then perform a Metropolis Markov Chain Monte Carlo (MCMC) simulation to obtain 5000 samples from the posterior distribution given in Eq. (25). The algorithm used is presented in detail in sec. **??** of Supplementary Information. The choice of α_d and proposal moves result in acceptance rates of $\approx 40\%$. The acceptance rates are summarized in table **??** of SI.

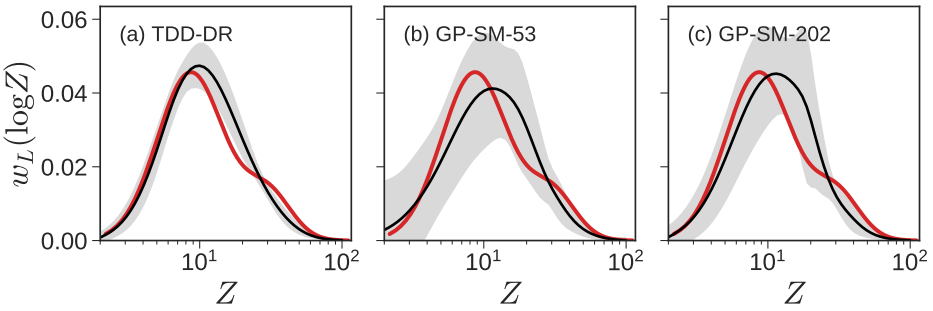


Fig. 11. The true MWD obtained from $\boldsymbol{x}_o$ is shown by the solid red line. The MWDs corresponding to the average of MCMC simulations with (a) TDD-DR, (b) GP-SM-53, and (c) GP-SM-202 (solid black lines) are also shown. The shaded gray region corresponds to the uncertainty of the inferred MWD.

Figure 10a compares the target spectrum ($\boldsymbol{h}_o$), and the mean prediction ($\hat{\boldsymbol{h}}_*$) of 10% of the MCMC samples using the TDD-DR as the forward model. Figures 10a and 10c repeat this calculation, using GP-SMs with $n = 53$ and 202 rather than TDD-DR as the underlying model. Samples appear to straddle the target spectrum as expected. The spectra sampled by the GP-SM-53 are qualitatively similar to the (true) TDD-DR model, and quantitatively similar to the more complex GP-SM-202. The cost of generating one MCMC sample for GP-SM-53 is under 0.2 sec, which is 3.5x smaller than the comparable cost for GP-SM-202.

Finally, figure 11 compares the true molecular weight distribution (MWD) $w_L(\log Z)$ with the mean prediction from the MCMC samples, and the associated error bar. To generate the inferred mean MWD curves, we constructed the MWD for each MCMC sample, and averaged over all of them. The shaded region represents the standard deviation computed over all the MCMC samples. Interestingly, the all three inverse models entertain the idea of a smooth unimodal MWD leading to $\boldsymbol{h}_o$. Perhaps the smooth form (single observable maxima) of $\boldsymbol{h}_o$ favors this "simpler" interpretation. All three curves miss the high chain length shoulder in the true MWD corresponding to $\boldsymbol{x}_o$.

TDD-DR, and to some extent GP-SM-202 capture the low molecular weight part of the MWD rather well. The mean predictions of all three inferred MWDs underestimate the fraction of high molecular weight chains. The uncertainty associated with the inferred MWD is lowest for TDD-DR. Nevertheless, if the true model establishes the best we can expect, the performance of the GP-SMs is quite satisfactory.

4. Summary and Conclusions

We explored a GP-based SM for describing the relaxation spectra of binary blends, and illustrated its use in inverse modeling, where the MWD of an unknown sample is inferred from a measured spectrum. We used a combination of the TDD-DR model and the program pyReSpect to obtain the spectrum $h(s)$ from a 5-dimensional input vector $\boldsymbol{x}$ that characterizes binary blends. We used this true model to generate training data of different sizes between $n = 30-1600$ via Latin hypercube sampling. The hyper-parameters $\boldsymbol{\theta}$ of a GP-SM with a separable covariance structure were obtained using the training data and MLE. The GP-SM interpolates the training data by design, and offers reasonable predictions of relaxation spectra with uncertainty estimates for other test binary blends.

The magnitude of the error as measured by the RMSE and MAD varied between one and two orders of magnitude smaller than the magnitude of the relaxation spectra depending on the training data size. Analysis of D_α suggests that scaled deviations of GP-SMs follow a PDF that is best described by a sum of two Cauchy distributions. These PDFs, while symmetric, are narrower than unit normal distributions near the mean, but also have fatter tails.

The seven hyper-parameters of the GP-SM obtained by MLE are relatively insensitive to the size of the training data n. This is fortunate because training a GP-SM, even with the separable covariance assumption, is an order $\mathcal{O}(N^3)+\mathcal{O}(n^3)$ operation. From this observation it is possible to reduce the number of iterations required to train the GP-SM for large n, by using the optimized parameters from a small n calculation as an initial guess. Both the MAD and RMSE decrease as the size of the training data increase, although a large number of outliers persist even at $n = 1606$. The shape of D_α versus α is preserved, although the tails become thinner as n increases beyond 100. In general, the accuracy of the GP-SM models increases as n is increased.

Finally, we considered the problem of inferring the MWD from synthetic data $\boldsymbol{h}_o$, which was generated by running the TDD-DR model on a known sample, $\boldsymbol{x}_o$. A MCMC-based sampling method was used to characterize the distribution of binary blends that are roughly consistent with the synthetic spectrum. Despite the modest difference in sampled spectra (figure 10), the MWD suggested (figure 11) by both the TDD-DR and GP-SM based models are surprisingly similar. In general, GP-SMs can be several orders of magnitude cheaper, which provides validation for using SMs for inverse problems in rheology. Interestingly, no big difference is observed between GP-SM-53 and GP-SM-202, although the generality of this finding needs to be further tested.

Acknowledgments

This work is based in part upon work supported by the National Science Foundation under grant no. NSF DMR-1727870.

References

Azzi, S. *et al.* (2019). Surrogate modeling of stochastic functions - application to computational electromagnetic dosimetry, *Int. J. Uncertain. Quantif.*, **9**: 351–363.

Bayarri, M. J. *et al.* (2007). Computer model validation with functional output, *Ann. Statist.*, **35**: 1874–1906.

Bhosekar, A. and Ierapetritou, M. (2018). Advances in surrogate based modeling, feasibility analysis, and optimization: A review, *Comput. Chem. Eng.*, **108**: 250–267.

Binder, K. (1986). *Monte Carlo and Molecular Dynamics Simulations in Polymer Science*, Oxford University Press, Oxford: UK.

Bird, R., Armstrong, R. and Hassager, O. (1987). *Dynamics of Polymeric Liquids, Volume 1: Fluid Mechanics*, John Wiley& Sons, New York.

Bird, R. and Giacomin, A. (2016). Polymer fluid dynamics: Continuum and molecular approaches, *Annu. Rev. Chem. Biomol. Eng.*, **7**: 479–507.

Carmesin, I. and Kremer, K. (1988). The bond fluctuation method - a new effective algorithm for the dynamics of polymers in all spatial dimensions, *Macromolecules*, **21**: 2819–2823.

Conti, S. and O'Hagan, A. (2010). Bayesian emulation of complex multi-output and dynamic computer models, *J. Stat. Plan. Inference.*, **140**: 640–651.

de Cloizeaux, J. (1988). Double reptation vs. simple reptation in polymer melts, *Europhy. Lett.*, **5**: 437–442.

de Cloizeaux, J. (1990). Relaxation of entangled polymers in melts, *Macromolecules*, **23**: 4678–4687.

de Gennes, P. G. (1979). *Scaling Concepts in Polymer Physics*, 1^{st} edition, Cornell University Press, Ithaca.

Dealy, J. M. and Larson, R. G. (2006). *Molecular Structure and Rheology of Molten Polymers*, 1^{st} edition, Hanser Publications.

Deng, S. *et al.* (2020). Comparative studies of surrogate models for response analysis of mistuned bladed disks, *Int. J. Comput. Methods*, **17**: 2050012.

Doi, M. and Edwards, S. F. (1986). *The Theory of Polymer Dynamics*, Clarendon Press, Oxford.

Frangos, M. *et al.* (2010). *Surrogate and Reduced-Order Modeling: A Comparison of Approaches for Large-Scale Statistical Inverse Problems*, chapter 7, John Wiley & Sons, Ltd, 123–149.

Fricker, T. E., Oakley, J. E. and Urban, N. M. (2013). Multivariate gaussian process emulators with nonseparable covariance structures, *Technometrics*, **55**: 47–56.

Furrer, R., Genton, M. G. and Nychka, D. (2006). Covariance tapering for interpolation of large spatial datasets, *J. Comput. Graph. Stat.*, **15**: 502–523.

Gneiting, T. (2002). Compactly supported correlation functions, *J. Multivar. Anal.*, **83**: 493–508.

Goldstein, M. and Rougier, J. (2006). Bayes linear calibrated prediction for complex systems, *J. Am. Stat. Assoc.*, **101**: 1132–1143.

Gramacy, R. B. and Apley, D. W. (2015). Local gaussian process approximation for large computer experiments, *J. Comput. Graph. Stat.*, **24**: 561–578.

Hensman, J., Fusi, N. and Lawrence, N. D. (2013). Gaussian processes for big data, in *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, UAI'13, AUAI Press, Arlington, Virginia, USA, 282–290.

Higdon, D. *et al.* (2008). Computer model calibration using high-dimensional output, *J. Am. Stat. Assoc.n*, **103**: 570–583.

Huang, S. P., Quek, S. T. and Phoon, K. K. (2001). Convergence study of the truncated karhunen–loeve expansion for simulation of stochastic processes, *Int. J. Numer. Methods. Eng.*, **52**: 1029–1043.

Jain, A. (1976). A fast karhunen-loeve transform for a class of random processes, *IEEE Trans. Commun.*, **24**: 1023–1029.

Janzen, J. and Colby, R. H. (1999). Diagnosing long-chain branching in polyethylenes, *J. Mol. Struct.*, **486**: 569–584.

Kaufman, C. G., Schervish, M. J. and Nychka, D. W. (2008). Covariance tapering for likelihood-based estimation in large spatial data sets, *J. Am. Stat. Assoc.*, **103**: 1545–1555.

Kennedy, M. and O'Hagan, A. (2000). Predicting the output from a complex computer code when fast approximations are available, *Biometrika*, **87**: 1–13.

Kennedy, M. C. and O'Hagan, A. (2001). Bayesian calibration of computer models, *J. R. Stat. Soc. Series B Stat. Methodol.*, **63**: 425–464.

Kremer, K. and Grest, G. S. (1990). Dynamics of entangled linear polymer melts - a molecular-dynamics simulation, *J. Chem. Phys.*, **92**: 5057–5086.

Kröger, M. and Hess, S. (2000). Rheological evidence for a dynamical crossover in polymer melts via nonequilibrium molecular dynamics, *Phys. Rev. Lett.*, **85**: 1128–1131.

Larson, R. G. (1988). *Constitutive equations for polymer melts and solutions*, Butterworth, Stoneham, MA.

Larson, R. G. *et al.* (2007). Advances in modeling of polymer melt rheology, *AIChE J.*, **53**: 542–548.

Li, G. *et al.* (2016). A hybrid reliability-based design optimization approach with adaptive chaos control using kriging model, *Int. J. Comput. Methods*, **13**: 1650006.

Likhtman, A. E. (2005). Single-chain slip-link model of entangled polymers: Simultaneous description of neutron spin-echo, rheology, and diffusion, *Macromolecules*, **38**: 6128–6139.

Likhtman, A. E., Sukumaran, S. K. and Ramirez, J. (2007). Linear viscoelasticity from molecular dynamics simulation of entangled polymers, *Macromolecules*, **40**: 6748–6757.

Álvarez, M. A., Rosasco, L. and Lawrence, N. D. (2012). Kernels for vector-valued functions: A review, *Found. Trends Mach. Learn.*, **4**: 195–266.

Masubuchi, Y. (2014). Simulating the flow of entangled polymers, *Annu. Rev. Chem.*

Biomol. Eng., **5**: 11–33, pMID: 24498953.

Masubuchi, Y. *et al.* (2008). Comparison among slip-link simulations of bidisperse linear polymer melts, *Macromolecules*, **41**: 8275–8280.

McFarland, J. *et al.* (2008). Calibration and uncertainty analysis for computer simulations with multivariate output, *AIAA Journal*, **46**: 1253–1265.

McKay, M. D., Beckman, R. J. and Conover, W. J. (1979). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code, *Technometrics*, **21**: 239–245.

Mead, D. W. (1996). Component predictions and the relaxation spectrum of the double reptation mixing rule for polydisperse linear flexible polymers, *J. Rheol.*, **40**: 633–661.

Nguyen-Tuong, D., Peters, J. and Seeger, M. (2008). Local gaussian process regression for real time online model learning and control, in *Proceedings of the 21st International Conference on Neural Information Processing Systems*, NIPS'08, Curran Associates Inc., Red Hook, NY, USA, 1193–1200.

Park, C. and Apley, D. (2018). Patchwork kriging for large-scale gaussian process regression, *J. Mach. Learn. Res.*, **19**: 1–43.

Phoon, K., Huang, H. and Quek, S. (2005). Simulation of strongly non-gaussian processes using karhunen–loeve expansion, *Probabilistic Eng. Mech.*, **20**: 188–198.

Rasmussen, C. E. and Ghahramani, Z. (2001). Infinite mixtures of gaussian process experts, in *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, NIPS'01, MIT Press, Cambridge, MA, USA, 881–888.

Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*, The MIT Press, Cambridge, MA.

Read, D. J., Shivokhin, M. E. and Likhtman, A. E. (2018). Contour length fluctuations and constraint release in entangled polymers: Slip-spring simulations and their implications for binary blend rheology, *J. Rheol*, **62**: 1017–1036.

Santner, T. J. *et al.* (2018). *The design and analysis of computer experiments*, 2nd edition, Springer, New York, NY.

Schieber, J. D. and Andreev, M. (2014). Entangled polymer dynamics in equilibrium and flow modeled through slip links, *Ann. Rev. Chem. Biomol. Engg.*, **5**: 367–381.

Seeger, M. W., Williams, C. K. I. and Lawrence, N. D. (2003). Fast forward selection to speed up sparse gaussian process regression, in *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, eds. C. M. Bishop and B. J. Frey, volume R4 of *Proceedings of Machine Learning Research*, PMLR, 254–261, reissued by PMLR on 01 April 2021.

Shaffer, J. S. (1994). Effects of chain topology on polymer dynamics - bulk melts, *J. Chem. Phys.*, **101**: 4205–4213.

Shanbhag, S. (2010). Analytical rheology of blends of linear and star polymers using a Bayesian formulation, *Rheol. Acta*, **49**: 411–422.

Shanbhag, S. (2011). Analytical rheology of branched polymer melts: Identifying

and resolving degenerate structures, *J. Rheol.*, **55**: 177–194.

Shanbhag, S. (2012). Analytical rheology of polymer melts: State of the art, *Int. Sch. Res. Notices*, **2012**.

Shanbhag, S. (2019a). Fast slip link model for bidisperse linear polymer melts, *Macromolecules*, **52**: 3092–3103.

Shanbhag, S. (2019b). pyReSpect: A computer program to extract discrete and continuous spectra from stress relaxation experiments, *Macromol. Theory Simul.*: 1900005.

Shanbhag, S. (2020a). How many monodisperse fractions are required to discretize polydisperse polymers?, *Macromol. Theory Simul.*, **29**: 10.1002/mats.202000020.

Shanbhag, S. (2020b). Relaxation spectra using nonlinear Tikhonov regularization with a Bayesian criterion, *Rheol. Acta*, **59**: 509–520.

Snelson, E. and Ghahramani, Z. (2006). Sparse gaussian processes using pseudo-inputs, in *Proceedings of the Neural Information Processing Systems Conference*, volume 18, MIT Press, British Columbia, Canada, 1257–1264.

Snelson, E. and Ghahramani, Z. (2007). Local and global sparse gaussian process approximations, in *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, eds. M. Meila and X. Shen, volume 2 of *Proceedings of Machine Learning Research*, PMLR, San Juan, Puerto Rico, 524–531.

Subramanian, G. and Shanbhag, S. (2008). On the relationship between two popular lattice models for polymer melts, *J. Chem. Phys.*, **129**: 144904.

Takeh, A. and Shanbhag, S. (2013). A computer program to extract the continuous and discrete relaxation spectra from dynamic viscoelastic measurements, *Appl. Rheol*, **23**: 24628.

Takeh, A., Worch, J. and Shanbhag, S. (2011). Analytical rheology of Metallocene-Catalyzed polyethylenes, *Macromolecules*, **44**: 3656–3665.

Tan, M. H. (2018). Gaussian process modeling of a functional output with information from boundary and initial conditions and analytical approximations, *Technometrics*, **60**: 209–221.

Tan, M. H. Y. (2019). Gaussian process modeling of finite element models with functional inputs, *SIAM-ASA J. Uncertain.*, **7**: 1133–1161.

Titsias, M. (2009). Variational learning of inducing variables in sparse gaussian processes, in *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, eds. D. van Dyk and M. Welling, volume 5 of *Proceedings of Machine Learning Research*, PMLR, Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA, 567–574.

Tsenoglou, C. (1987). Viscoelasticity of binary homopolymer blends, in *ACS Polym. Preprints*, volume 28, 185–186.

Tsenoglou, C. (1991). Molecular-weight polydispersity effects on the viscoelasticity of entangled linear-polymers, *Macromolecules*, **24**: 1762–1767.

Tzoumanekas, C. and Theodorou, D. N. (2006). From atomistic simulations to slip-link models of entangled polymer melts: Hierarchical strategies for the prediction

of rheological properties, *Curr. Opin. Solid State. Mater. Sci.*, **10**: 61–72.

van Ruymbeke, E., Liu, C.-Y. and Bailly, C. (2007). Quantitative tube model predictions for the linear viscoelasticity of linear polymers, *Rheology Reviews*: 53–134.

van Ruymbeke, E. *et al.* (2002). Evaluation of reptation models for predicting the linear viscoelastic properties of entangled linear polymers, *Macromolecules*, **35**: 2689–2699.

Vanhatalo, J. and Vehtari, A. (2008). Modelling local and global phenomena with sparse gaussian processes, in *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence (UAI-08)*, eds. D. McAllester and P. Myllymäki, volume 2008, AUAI Press, International, 571–578, proceeding volume: 2008; null ; Conference date: 09-07-2008 Through 12-07-2008.

Wasserman, S. H. and Graessley, W. W. (1992). Effects of polydispersity on linear viscoelasticity in entangled polymer melts, *J. Rheol.*, **36**: 543–572.

Watanabe, H. *et al.* (2004). Test of full and partial tube dilation pictures in entangled blends of linear polyisoprenes, *Macromolecules*, **37**: 6619–6631.