Gaussian processes for efficient numerical simulations in physics
ORIE 6741 final project midterm report

Christophe Bonneville (cpb97@cornell.edu) [*1] **Maxwell Jenquin (mrj89@cornell.edu)** [*1]

[1]Cornell University, Ithaca NY, USA [*]Equal Contribution

1. Introduction

Many physical phenomena, such as heat transfer, fluid behavior, and quantum mechanics, are governed by partial differential equations (PDE). As a result solving these equations accurately over a wide array of space and time domains is of critical interest to scientific and engineering applications. However, as is well known, there is no general method to solve PDE, and analytic solutions exist for only very few and usually simplistic problems. To address this issue, mechanistic numerical analysis methods such as finite elements and numerical integration schemes have been used to find approximate solutions. These methods have had tremendous success in the last 30 years and are widely used for simulations in physics and engineering. However, they are computationally expensive (complex or high-resolution simulations can take days to run) and some equations remain difficult to approximate. More efficient approaches which retain accuracy are clearly needed and would be a breakthrough in applied physics fields.

Machine learning provides a potential framework for avoiding the expensive computation required by mechanistic methods. In particular, recent papers (Umetani & Bickel, 2018; Raissi & Karniadakis, 2018) suggest that Gaussian processes (GPs, the theory of which we will not discuss here - refer to Rasmussen & Williams (2005)) can be used along with somewhat sparse results from simulations or measurements to predict key traits of related systems, or learn parameters from small snapshots of known systems. Even for the Navier-Stokes equation, these papers find that high-quality results can be recovered and interpolated using GP-based techniques. Additionally, GPs are inherently well-suited to problems of this nature because datasets are limited in size by computational expense. The confidence intervals resulting from such techniques are also useful tools for estimating the reliability of a solution (which is critical in engineering applications).

This project proposes to extend and improve the approaches of these papers. In section 2 we develop and present preliminary interpolation strategies for domain and boundary condition parameters. In section 3 we develop and test a method for time extrapolation of PDE systems.

2. Interpolating Domain and Boundary Conditions

2.1. Related Work

2.1.1. Standard Methods for Steady-State PDE Approximation

To develop our method in this portion of the project we focus on the heat equation, a linear PDE describing the temperature behavior inside a domain Ω, given arbitrary heat input q and prescribed temperatures $\bar{u}$ at the boundaries of Ω. The heat equation for steady-state (time independent) 2D problems is given by:

$$\begin{cases} \dfrac{\partial^2 u}{\partial^2 x} + \dfrac{\partial^2 u}{\partial^2 y} = q, & (x,y) \in \Omega \subset \mathbb{R}^2 \\ u : \Omega \to \mathbb{R}, & u = \bar{u} \quad \text{for} \quad (x,y) \in \partial\Omega \end{cases} \tag{1}$$

To generate data, we have solved this equation approximately using the finite element method (FEM), as described by Hughes (1987). The core concept of FEM is to approximate the solution on a set of n points throughout Ω. Doing so requires creating a mesh of the domain, which takes up to $\mathcal{O}(n^2)$ computations (Ruppert, 1995). Then, we can find the approximate values of the solution by solving a linear system with respect to $\tilde{u}$, which is the discretized approximate solution vector (size n). In general this matrix inversion requires $\mathcal{O}(n^3)$ operations, as usual. Although FEM is a very popular technique in engineering and physics, it is time consuming due to mesh creation and matrix inversion ($\mathcal{O}(n^2)$ and $\mathcal{O}(n^3)$, note that a finer approximation requires a higher n). In engineering design processes, we often need to re-iterate simulations to obtain a satisfactory design, leading to potentially redundant computations. We are also usually interested in the extrema of $\tilde{u}$ rather than $\tilde{u}$ itself, which motivates a more efficient means of extracting an estimate of that value (rather than going through the full process of FEM simulation).

2.1.2. Existing Work Towards Solution Prediction

Despite advances in parallel computing, the drawbacks of FEM techniques described above are still inherent to the

approach. Machine learning techniques are a promising tool for improving efficiency in that respect, but one major issue standing in the way has historically been domain parameterization - in order to be suitable inputs for machine learning techniques, a wide range of domains must be described by a few consistent parameters. This issue has been partly addressed by Umetani & Bickel (2018), with the development of a novel parametrization method to describe complex geometries with a systematic number of features. Subsequently, they created a dataset of FEM simulation results for the flow of air around car bodies, on which they trained a Gaussian Process with an ARD kernel. They were able to obtain fair flow predictions for interpolant car models with relatively little computation time. Their work is a valuable proof of concept and shows that predicting PDE solutions with Gaussian processes is feasible and efficient. In this portion of the project we propose to extend their work but with simpler parameterizations by finding a framework for linear equation predictions using an expressive kernel.

2.2. Interpolation Model

Following the work of Umetani & Bickel (2018), we used an FEM simulation to generate a dataset of results for the heat equation in a rectangular domain $\Omega = a \times b \subset \mathbb{R}^2$ with an intermediate level of discretization. For each simulation, four input features were randomly drawn from a uniform distribution: the length and width of the domain (a and b), the scalar heat input q (with units of temperature per area), and the boundary condition $\bar{u}$ (with units of temperature). Hence the input feature vector is $\mathbf{x}_i = (a, b, q, \bar{u})$ and the target is $y_i = \max\{\tilde{u}\}$ (maximum temperature within Ω). Note that each input feature has values within the range $[0, 100]$, and the target variables lie in the range $y_i \in [5, 6.3 \times 10^4]$.

Although existing work uses an ARD kernel, we can exploit the linear nature of the heat equation for better prediction. Indeed, if one of the features change while the other three remain fixed, the maximum temperature varies linearly. Therefore, we base our model on the Ornstein-Ulhenbeck kernel which acts as a strong pointwise interpolator and is robust to noise (quite surprisingly however, a squared Ornstein-Ulhenbeck kernel tends to provide slightly better results). Due to the expected monotonicity of our target variable with respect to domain size, heat input and boundary conditions, we use a combination of non-stationary linear kernels to capture this monotonic trend and extrapolate beyond the span of the dataset. As a result, our proposed kernels are respectively, for interpolation and extrapolation:

$$k_{\text{interp}}(x, x') = \theta_i k^2_{\text{OU}}(x, x') \tag{2}$$

$$k_{\text{extrap}}(x, x') = \sum_{i=1}^{4} \theta_i k^2_{\text{OU},i}(x, x') k_{\text{LIN},i}(x, x') \tag{3}$$

Note that in equation 3 each component OU kernel depends on three variables and the corresponding linear kernel depends on the remaining variable

2.3. Results and Discussion

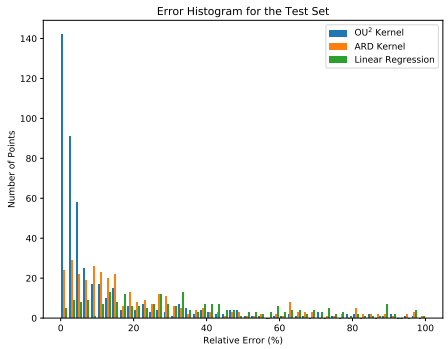


Figure 1. Relative error histogram between the predicted maximum temperature and the test set for rectangular domains with the OU squared kernel, the ARD kernel and the standard linear regression

We train our interpolation model using 500 data points. Figure 1 shows the relative error between our prediction and the test set which contains 500 points. Clearly, the squared Ornstein-Ulhenbeck kernel outperforms both the ARD kernel and a standard multi-linear regression technique. However, at this point in the project the method of extrapolation is not yet fully functional, so results are held for later presentation.

Next we apply our interpolation model to more complex domains with polygonal geometries. Following the same approach as with rectangular domains, we created a new data set for random quadrilaterals. Each quadrilateral is fully described by the coordinates of its vertices listed in counter clockwise order, yielding an input vector with 10 features of

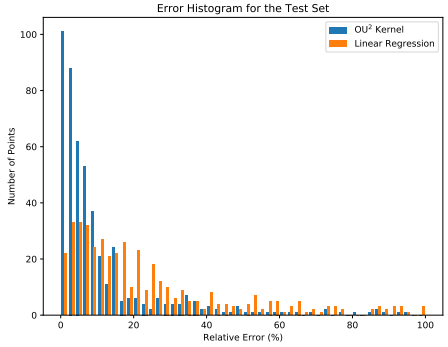


Figure 2. Relative error histogram between the predicted maximum temperature and the test set for quadrilateral polygons with the OU squared kernel and the standard linear regression

the form $\mathbf{x}_i = (x^1, y^1, ..., x^4, y^4, q, \bar{u})$. Again, the training set and the test set each contain 500 points. As shown in figure 2, the squared Ornstein-Ulhenbeck kernel still provides satisfactory predictions even with a higher number of features, meaning it has the potential to be extended to more complicated geometries and sophisticated domain parameterizations.

3. Extrapolating in Time

3.1. Introduction

Time-dependent partial differential equation (PDE) systems govern the majority of fundamental physical processes and phenomena, but are often difficult to describe and simulate with numerical methods. Even more challenging perhaps is identifying (and extrapolating the dynamics of) a PDE system given a small number of measurements of its behavior. In pursuit of this goal, Raissi & Karniadakis (2018) have proposed and demonstrated a method for learning the parameters of a homogeneous PDE system using linearization of the differential operator and Gaussian process (GP) regression, via so-called numerical Gaussian processes (Raissi et al. (2018)). In this section of the project we use and modify their model to explore the extrapolation of dynamics of a semi-undiscovered PDE system, as a potential alternative to standard methods of integration when PDE parameters are not known.

3.2. Related Work

3.2.1. Standard Methods for PDE Integration

If the PDE which governs a time-dependent system is explicitly defined, there exist many standard methods for integration, most of which are specified to the particular type of PDE involved in the problem. In addition to time-dependent finite-element methods related to those described in section 2, finite-difference methods such as Crank-Nicolson (LeVeque, 2007) are popular choices for certain problems due to properties like energy conservation (for elliptic PDE) and efficiency for certain linear problems. For some problems, spectral methods (Kopriva, 2009) are employed, resulting in a similar computational cost to finite element methods.

All of these methods suffer from some form of matrix inversion as a requirement, therefore making them $\mathcal{O}(n^3)$ without certain problem-dependent special structure. In addition they fix the domain discretization for the length of the entire simulation, and generally restrict that discretization to some level of detail to ensure convergence. Because of this there is room for Gaussian process-based approaches to be not only strong predictors in the case where the system is not precisely known, but also to compete when the system is known exactly. Modern advancements in inducing-point methods such as structured kernel interpolation (Wilson & Nickish, 2015) also provide the potential for future work on GP extrapolation to be more efficient than standard methods for certain tasks.

3.2.2. Existing Work Towards PDE Discovery

Following Raissi and Karniadakis (2018), suppose we have a PDE system on a 1-D domain governed by the following equation, where $\mathcal{L}_x$ is a linear differential operator which contains only spatial derivatives. If a PDE has a nonlinear operator we assume there exists an appropriate linearization and proceed.

$$\frac{\partial}{\partial t}u(x,t) = \mathcal{L}_x u(x,t) \tag{4}$$

Further suppose that we measure the system at some points along its domain twice, with measurements close enough in time that we may approximate the evolution of the system via the backwards Euler finite difference equation,

$$u_n \approx u_{n-1} + \Delta t \mathcal{L}_x u_n. \tag{5}$$

Here we will use the notation $u_j = u(x, t_j)$. Then by placing a GP prior on u_n we use equation 5 to induce a GP prior on u_{n-1}, since the two are related by a linear operator and any linear transformation applied to a GP results in a GP:

$$u_n \sim \mathcal{GP}(0,k) \Rightarrow u_{n-1} \sim \Delta t(I - \mathcal{L}_x)\mathcal{GP}(0,k) \tag{6}$$

where I is the identity. This induces a joint GP distribution over our two measurements,

$$\begin{bmatrix} u_n \\ u_{n-1} \end{bmatrix} \sim \mathcal{GP}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} k & k_{1,2} \\ k_{2,1} & k_{2,2} \end{bmatrix} \right) \tag{7}$$

where we have defined

$$k_{1,2}(x,x') = \Delta t(I - \mathcal{L}_x)k(x,x') \tag{8}$$
$$k_{2,1}(x,x') = \Delta t(I - \mathcal{L}_{x'})k(x,x') \tag{9}$$
$$k_{2,2}(x,x') = (\Delta t)^2(I - \mathcal{L}_{x'})(I - \mathcal{L}_x)k(x,x') \tag{10}$$

Existing work uses this joint distribution, along with an RBF kernel as $k(x,x')$ with analytic derivatives to define the other sub-kernels, to learn the coefficients of $\mathcal{L}_x$ as hyperparameters of the joint kernel, given the functional form of the linear operator.

3.2.3. The Spectral Mixture Kernel

Introduced in 2013 by Wilson et. al (Wilson & Adams, 2013), the spectral mixture kernel approximates any stationary kernel via a mixture of Gaussian distributions in its spectral density. In one dimension it is given by

$$k(\tau) = \sum_{q=1}^{Q} w_q \exp(-2\pi^2\tau^2\nu_q)\cos(2\pi\tau\mu_q) \tag{11}$$

where $\tau = x - x'$, and the spectral density of k is a mixture of Q Gaussians, with means μ_q and variances ν_q, weighted by w_q. In addition it has reasonably simple analytic derivatives, which makes it convenient for use in the framework described by Raissi and Karniadakis.

3.3. Extrapolation Model

In existing work, the dynamics of a PDE system have been encoded into a GP model and trained. But, no attempt to explore the use of that model to simulate dynamics has been made. Here we extend and develop the method in section 3.2.2 to extrapolate the dynamics of the discovered PDE system. After measuring u_n and u_{n-1} and training the joint GP model described, we treat the joint distribution between u_n and u_{n+1} (where $t_{n+1} - t_n = t_n - t_{n-1}$) as a predictive distribution for u_{n+1}(Rasmussen & Williams, 2005). That is, with trained kernel functions, our predictive distribution yields:

$$\begin{bmatrix} u_{n+1} \\ u_n \end{bmatrix} \sim \mathcal{GP}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} k & k_{1,2} \\ k_{2,1} & k_{2,2} \end{bmatrix} \right) \tag{12}$$

$$\Rightarrow \mathbb{E}[u_{n+1}] = k_{1,2}(x_n, x_{n+1}) \Big(k(x_n, x_n) + \sigma I \Big)^{-1} u_n \tag{13}$$

where σ is the learned noise variance of the posterior model, x_n is the set of sampled locations at the second measurement and x_{n+1} is the set of x values at which we would like to predict the solution into the future. This process is repeated iteratively, evaluating the predictive distribution for the induced joint GP posterior over u_{n+1} and u_{n+2}, and onwards into the future.

In initial tests we employ Raissi and Karniadakis' model for integration, which uses an RBF kernel as $k(x, x')$, chosen largely for its properties as a universal approximator and simple analytic derivatives. All results presented in this midterm report use the RBF kernel model as a proof of concept. However in the remainder of the project we will extend the model to use the more expressive spectral mixture kernel described in section (whatever), in order to better capture the true dynamics of the systems analyzed.

3.4. Results and Discussion

Preliminary evaluation of previous work reveals that the fit of Raissi and Karniadakis' model to the parameters of the PDE depend somewhat heavily on the sampled timestep of their high-fidelity simulation (figure 3). As such, care was taken to choose an appropriate sample point for reported results so as to demonstrate a "best case" situation for each equation analyzed with this method. In the rest of this section, we choose a convenient high-quality fit sample point. Simulations were provided by Raissi and Karniadakis as sample data for their paper.

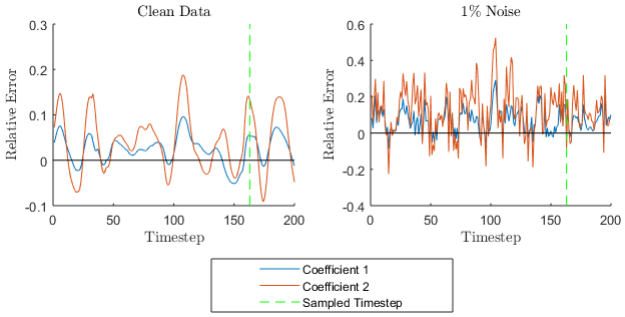
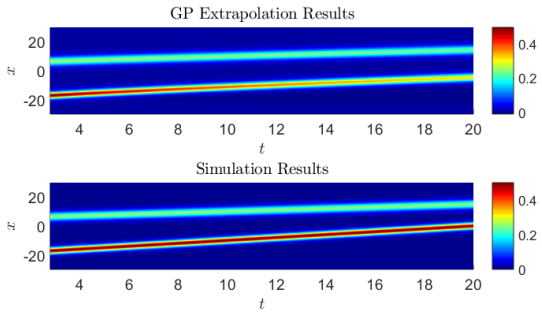


Figure 3. Relative error in model parameters found by Raissi and Karniadakis' method as a function of sample time for the Kortweg-de Vries equation, clean data (left) and 1% noise (right). Note the difference in magnitude for the two plots. On each plot the vertical line is the timestep sampled in Raissi & Karniadakis (2018).

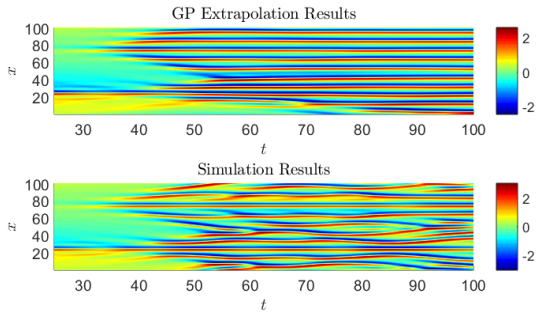


Governing Equation: $\dfrac{\partial u}{\partial t} = -6u\dfrac{\partial u}{\partial x} - \dfrac{\partial^3 u}{\partial x^3}$

Figure 4. Extrapolation results (top) for the Kortweg-de Vries model compared with simulation results (bottom).

In figures 4, 5 and 6 we report the quality of extrapolation as compared with the high-fidelity numerical simulations used as ground truth. These three examples illustrate the qualitative success of the extrapolation model, while also indicating the three apparent issues with the approach. In the context of the Kortweg-de Vries equation (figure 4), we observe a classic two-soliton solution in which the two peaks approach each other with the correct velocity. However, because the backwards Euler finite difference formula is not energy conserving, the GP solution leaks energy from the larger peak over time. Depending on the equation in play, this can lead to qualitatively different solutions, and motivates future work in the direction of symplectic integration schemes to replace the backwards Euler means of inducing a GP prior.

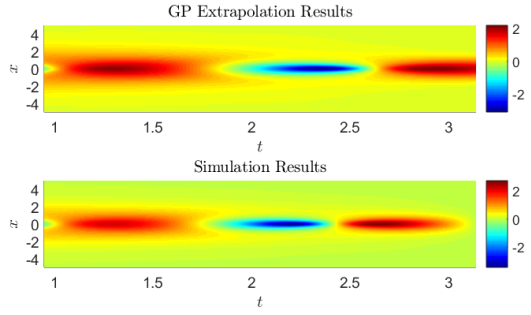
In the case of the Kuramoto-Sivashinsky equation (figure 5), the system at hand begins with a single peak which explodes



Governing Equation: $\frac{\partial u}{\partial t} = -u\frac{\partial u}{\partial x} - \frac{\partial^2 u}{\partial x^2} - \frac{\partial^4 u}{\partial x^4}$

Figure 5. Extrapolation results (top) for the Kuramoto-Sivashinksy model compared with simulation results (bottom).

into chaos throughout the domain, behavior which has been well-studied (Smyrlis & Papageorgiou, 1991). The onset of this chaos is well-approximated by the GP model, a very encouraging result. However, in the simulation we see that the peaks interact in a nonlinear fashion and are not stationary, while in the GP model they remain largely static after settling. This is explained by the linearization of the system, which changes the intricate dynamics of this chaotic state significantly. Still, in any practical application the pattern of chaos onset is a quantity of interest, and so our extrapolation model has utility in this context. Unfortunately, the requirement for linearity of the differential operator is strict due to the nature of GPs, therefore this hurdle is not one that can be overcome in this theoretical context.



Governing Equation: $i\frac{\partial u}{\partial t} = -\frac{1}{2}\frac{\partial^2 u}{\partial x^2} - |u|^2 u,\ u \in \mathbb{C}$

Figure 6. Extrapolation results (top) for the nonlinear Schrodinger equation compared with simulation results (bottom). Due to space considerations only the real part of the solution is plotted.

For the nonlinear Schrodinger equation (figure 6), we observe that the core behavior of the system, namely periodicity of the single wave solution, is largely preserved. However the period itself is sensitive to the parameters of the equation, and so a qualitative change in dynamics is seen based on small approximation errors no matter how well-chosen the sample point may be. Sensitivity of model behavior to parameter settings is another insurmountable hurdle, one shared by all methods of integration. Issues such as this one motivate the introduction of a more sophisticated kernel which will improve the result of optimization of hyperparameters.

While the paper by Raissi and Karniadakis evaluated the effect of noise on the fit to the PDE model in each case, we found that the solution quality for extrapolation did not differ significantly from the noiseless case in each system tested.

With these results we have demonstrated that while the GP extrapolation technique described in section 3.3 may not produce physically precise descriptions of the evolution of a PDE system, qualitative dynamics and key features were well reproduced in several contexts. With flexibility in choice of domain sample points from timestep to timestep as well as the potential for efficient implementation using inducing point methods or structured kernel interpolation, GP integration becomes a potentially attractive means of approximating key features of a system with unknown coefficients.

4. Future Work

In the area of domain and boundary condition interpolation, goals for the remainder of the project are to improve our prediction accuracy by adding more data points in the error-prone regions of parameter space, and extend our approach to higher order polygon domains and advanced boundary conditions. Finally we will provide an analysis of the computational expense of this technique as compared to FEM simulations, and finalize and test our extrapolation model. Future work will involve applying this concept to other linear PDE.

With respect to time extrapolation of PDE systems, remaining goals for this project are the implementation of the spectral mixture kernel in the existing framework, and a careful analysis of performance between the two kernel choices. Potential future directions of investigation include the development of this technique using an energy-conserving numerical integration scheme, and the exploration of efficient means of approximating these predictive distribution evaluations, via structured kernel interpolation or inducing point methods.

References

Hughes, T. J. *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*. Dover Publications, 1987.

Kopriva, D. *Implementing Spectral Methods for Partial Differential Equations*. Springer, Dordrecht, 2009.

LeVeque, R. J. *Finite Difference Methods for Ordinary and Partial Differential Equations*. SIAM, Philadelphia, PA, 2007.

Raissi, M. and Karniadakis, G. E. Hidden physics models: Machine learning of nonlinear partial differential equations. *Journal of Computational Physics*, 357, 2018.

Raissi, M., Perdikaris, P., and Karniadakis, G. E. Numerical gaussian processes for time-dependent and non-linear partial differential equations. *SIAM J. Sci. Computation*, 40(1), 2018.

Rasmussen, C. E. and Williams, C. K. I. *Gaussian Processes for Machine Learning*. The MIT Press, 2005.

Ruppert, J. A delaunay refinement algorithm for quality 2-dimensional mesh generation. *Journal of Algorithms*, 18(3), 1995.

Smyrlis, Y. and Papageorgiou, D. Predicting chaos for infinite dimensional dynamical systems: the kuramoto-sivashinsky equation, a case study. *Proceedings of the National Academy of Sciences*, 88(24), 1991.

Umetani, N. and Bickel, B. Learning three-dimensional flow for interactive aerodynamic design. *ACM Trans. Graph.*, 37(4), 2018.

Wilson, A. and Adams, R. Gaussian process kernels for pattern discovery and extrapolation. *International Conference on Machine Learning*, 2013.

Wilson, A. and Nickish, H. Kernel interpolation for scalable structured gaussian processes (kiss-gp). *International Conference on Machine Learning*, 2015.