General linear-time inference for Gaussian Processes on one dimension

Jackson Loper [1] **David Blei** [1] **John P. Cunningham** [1] **Liam Paninski** [1]

Abstract

Gaussian Processes (GPs) provide a powerful probabilistic framework for interpolation, forecasting, and smoothing, but have been hampered by computational scaling issues. Here we prove that for data sampled on one dimension (e.g., a time series sampled at arbitrarily-spaced intervals), approximate GP inference at any desired level of accuracy requires computational effort that scales *linearly* with the number of observations; this new theorem enables inference on much larger datasets than was previously feasible. To achieve this improved scaling we propose a new family of stationary covariance kernels: the Latent Exponentially Generated (LEG) family, which admits a convenient stable state-space representation that allows linear-time inference. We prove that any continuous integrable stationary kernel can be approximated arbitrarily well by some member of the LEG family. The proof draws connections to Spectral Mixture Kernels, providing new insight about the flexibility of this popular family of kernels. We propose parallelized algorithms for performing inference and learning in the LEG model, test the algorithm on real and synthetic data, and demonstrate scaling to datasets with billions of samples.

1. Introduction

Gaussian Process (GP) methods are a powerful and expressive class of nonparametric techniques for interpolation, forecasting, and smoothing. However, this expressiveness comes at a cost: if implemented naively, inference in a GP given m observed data points will require $O(m^3)$ operations. A large body of work has devised various means to circumvent this cubic run-time; briefly, this literature can be broken down into several threads. A first approach is to attempt to perform exact inference without imposing any restrictions on the covariance kernel, using careful numerical methods typically including preconditioned conjugate gradients (Cutajar et al., 2016). Wang et al. (2019) represents the state of the art: inference and learning can be performed on $\sim 10^6$ datapoints on an 8-GPU machine and a few days of processing time. A second approach searches for good approximations to the posterior that do not rely on special properties of the covariance kernel. Some well-known examples of this approach include (Quiñonero-Candela & Rasmussen, 2005; Snelson & Ghahramani, 2007; Hensman et al., 2013; Low et al., 2015; De G. Matthews et al., 2017). In a third approach, several techniques exploit special kernel structure. Examples include matrices with Kronecker product structure (Gilboa et al., 2013), Toeplitz structure (Zhang et al., 2005; Cunningham et al., 2008), matrices that can be well-approximated with hierarchical factorizations (Ambikasaran et al., 2015), or matrices which are sufficiently smooth to allow for interpolation-based approximations (Wilson & Nickisch, 2015).

When the GP has one-dimensional input – e.g., a scalar or vector time-series sampled at arbitrary time points – the most popular method for scaling learning and inference is to approximate the GP with some form of Gaussian hidden Markov model (that is, a state-space model) (Reinsel, 2003; Mergner, 2009; Cheung et al., 2010; Brockwell & Davis, 2013). This model class has considerable virtue: it is a particular case of a GP, it includes popular models like the auto-regressive moving average process (ARMA, when on an evenly spaced grid), and perhaps most importantly it admits linear-time $O(m)$ inference via message passing.

Is approximating a GP with a state-space model a generally viable strategy? In several special cases it has been shown that state-space models provide excellent approximations of specific covariance kernels (Karvonen & Sarkkä, 2016; Benavoli & Zaffalon, 2016). Discrete-time processes on a finite interval can also be approximated this way (Lindgren et al., 2011). Practically, (Gilboa et al., 2013) shows it is straightforward to learn many GP models using a state-space model.

In this work we establish the full generality of this strategy: we offer a new theorem proving that *any* GP on one dimension with a Lebesgue-integrable continuous kernel can be arbitrarily well approximated by a specifically-chosen state-space model. By doing so, we effectively reduce the

[1]Columbia University, New York, New York, USA. Correspondence to: Jackson Loper <jl5116@columbia.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Vienna, Austria, PMLR 108, 2020.

run-time burden of GPs on one dimension from cubic to linear.

We first develop a new class of Gaussian hidden Markov models on one dimension: the Latent Exponentially Generated (LEG) process. This model family is a generalization of the Celerite family of Gaussian Processes (Foreman-Mackey et al., 2017). Unlike some popular state-space models such as the ARMA, LEG processes do not require that the observations are equally spaced. These models define a distribution on vector-valued functions on the entire real line, $X : \mathbb{R} \to \mathbb{R}^n$. By construction, LEG processes are stable and stationary, with a kernel that can be evaluated easily, and inference requires linear time. In addition, we here show that inference for these models can be parallelized efficiently via a technique known as Cyclic Reduction (Sweet, 1974), leading to significant runtime improvements. Furthermore these models are general: our main mathematical result is to prove that for any stationary Gaussian Process X on one dimension with integrable continuous covariance, for any ε, the covariance of X can be matched within ε by a LEG covariance kernel.

LEG kernels generalize the Celerite kernel (Foreman-Mackey et al., 2017) by allowing more model flexibility and permitting vector-valued observations. Every Celerite kernel can be understood as a special case of a LEG kernel.

The remainder of this paper defines the LEG family, derives its essential properties and generality, and finally empirically backs up these claims across real and synthetic data. In particular, we show that the LEG family enables inference on datasets with billions of samples with runtimes that scale in minutes, not days.

2. Preamble: Gaussian process generalities

A Gaussian Process on one dimension is a random function $X : \mathbb{R} \to \mathbb{R}^n$ such that for any finite collections of times $t_1, t_2 \cdots t_m$ the joint distribution of $(X(t_1), \cdots X(t_m)) \in \mathbb{R}^{m\times n}$ is jointly Gaussian. The covariance kernel of X is a matrix-valued function defined by

$$\Sigma(s,t) = \mathrm{Cov}(X(s), X(t)) \in \mathbb{R}^{n\times n}.$$

A process is said to be stationary if $\Sigma(s,t) = C(s-t)$ for some matrix-valued function C and $\mathbb{E}[X(t)]$ is the same for all values of t. In this case we write τ for the time-lag $t-s$, i.e $C = C(\tau)$.

We will focus on two critical computational tasks here: inference and learning. "Inference" refers to using a GP model to compute the conditional densities of X given a finite collection of observations $D = (X(t_1), \cdots X(t_m))$. "Learning" refers to estimating the covariance of X from the data D. Both of these tasks can be computationally intensive: naive evaluation of the likelihood of D requires computing the determinant of an $m\times m$ matrix and solving an m-dimensional linear system. In general these tasks require $O(m^3)$ operations. Here we circumvent this scaling law by restricting ourselves to a parametric family of kernels which admit linear-time (i.e., $O(m)$) algorithms. We further show that this restriction is without loss of generality, since this family of kernels is capable of approximating any integrable continuous stationary kernel on the real line.

3. The LEG kernel

We introduce a parametric family of random processes on one dimension: the Latent Exponentially Generated (LEG) process. This process will achieve both goals of this work: linear-time inference and arbitrary approximation quality to any GP. For clarity of exposition, what follows assumes stationarity and zero mean; generalizations are discussed in Section 4. We first define the latent GP, after which we define the observation model; taken together these objects will form the LEG family.

In designing a family of latent GP models, our first goal is to enable fast computation: the models should be stationary, with an easily-computed kernel. In addition, it is convenient to focus on Markovian models, since the Markov property will enable efficient inference.

A general and classic family of Markovian models are given by linear Langevin equations (Coffey et al., 2004), i.e. processes defined by

$$z(t) = z(0) + \int_0^t (-Gz(s)ds + \sigma dw(s)),$$

where w is an ℓ-dimensional Brownian motion, and G, σ are square matrices. To ensure this process doesn't grow without bound, we need the real part of the eigenvalues of G to be positive. Guaranteeing this nontrivial constraint is challenging (Buesing et al., 2012; Choudhary et al., 2019). To remedy this problem we developed a closely-related family of models which are always stable and stationary:

Definition 1. Let $z(0) \sim \mathcal{N}(0, I)$, let w denote a Brownian motion, let N, R be any $\ell \times \ell$ matrices, and let $G = NN^\top + R - R^\top$. Let z satisfy

$$z(t) = z(0) + \int_0^t \left(-\frac{1}{2}Gz(s)ds + Ndw(s)\right).$$

Then we will say z is a Purely Exponentially Generated process, $z \sim \mathrm{PEG}(N, R)$.

This family has another advantage: the covariance kernel is easy to compute. The covariance kernel for linear Langevin models usually involves an integral (Vatiwutipong & Phewchean, 2019), but for PEG models we can compute this integral in closed form:

Lemma 1. *$z \sim \mathrm{PEG}(N, R)$ is stationary, with covariance kernel given by*

$$C_{\mathrm{PEG}}(\tau; N, R) \triangleq \exp\left(-\frac{\tau}{2}\left(NN^\top + R - R^\top\right)\right).$$

Proof: See supplementary material.

The matrices N, R can be interpreted intuitively. The positive definite diffusion $NN^\top$ controls the predictability of the process: when an eigenvalue of $NN^\top$ becomes larger, the process Z becomes less predictable along the direction of the corresponding eigenvector. The antisymmetric $R - R^\top$ term affects the process by applying an infinitesimal deterministic rotation at each point in time. The eigenvalues of $R - R^\top$ are purely imaginary, and when they are large they lead to rapid oscillations in the process, while the eigenvectors control how these oscillations are mixed across the dimensions of Z. As an illustration, Figure 1 shows the first dimension of samples from an $\ell = 2$ dimensional PEG process with various values of N, R.

We now turn to the observed process:

Definition 2. Let $z \sim \mathrm{PEG}(N, R)$. Fix any $n \times \ell$ matrix B and any $\ell \times \ell$ matrix Λ. For each t independently, define the conditional observation model:

$$x(t)|z(t) \sim \mathcal{N}(Bz(t), \Lambda\Lambda^\top).$$

We define a **Latent Exponentially Generated (LEG)** process to be the Gaussian Process $x : \mathbb{R} \to \mathbb{R}^n$ generated by a PEG prior and the above observation model. We write $x \sim \mathrm{LEG}(N, R, B, \Lambda)$. X has a LEG kernel:

$$C_{\mathrm{LEG}}(\tau; N, R, B, \Lambda) \triangleq B\left(C_{\mathrm{PEG}}(\tau; N, R)\right)B^\top + \delta_{\tau=0}\Lambda\Lambda^\top.$$

Here δ is the indicator function, and again $\tau > 0$. We will refer to the latent dimension ℓ as the **rank** of the LEG kernel.

3.1. Computation with LEG processes

The LEG model is a Gaussian hidden Markov model. As usual in such models, it follows that problems of evaluation, interpolation, smoothing, and sampling reduce to operations with block-tridiagonal matrices (De Jong, 1988). This block-tridiagonal structure is what enables linear-time inference.

While the obvious choice for processing these block-tridiagonal matrices might be a Kalman filter, it is not ideally suited for modern hardware: the naïve Kalman filter requires a single sequential sweep through the data. If the latent process has a quick mixing time this requirement can be relaxed (Gonzalez et al., 2009), but we seek an algorithm that parallelizes efficiently regardless of the parameters of the model.

We here propose to use Cyclic Reduction (CR) techniques instead. These offer a convenient parallelizable approach to computation with block-tridiagonal matrices (Sweet, 1974). To our knowledge, the CR approach has not previously been applied in the Gaussian Process literature. Like the Kalman filter, CR can be understood as a linear-time Cholesky decomposition algorithm for block-tridiagonal matrices (Eubank & Wang, 2002). Linear-time Cholesky decompositions lead directly to linear-time algorithms for solving linear systems and computing the determinant, which, in turn, allows us to compute all quantities required for inference in LEG processes. The difference between the Kalman filter and CR is "pivoting"; CR computes the Cholesky decomposition of a carefully permuted version of a block-tridagonal matrix. This pivoting allows the CR algorithm to proceed in $\log_2 m$ parallelizable stages, each stage concerning a matrix half the size of the matrix from the previous stage[1]. Unlike the Kalman Filter, CR can be completed with k processors on the order of m/k time (as long as $k < m$). Implementing parallel versions of CR in modern software libraries (TensorFlow2 in this case) was straightforward, making it easy to take advantage of modern hardware.

Exact linear-time algorithms for likelihood (and gradient) evaluation, smoothing, forecasting, and interpolation are given in the supplement. TensorFlow2-based Python code, tutorial notebooks, and API documentation can be found at `https://github.com/jacksonloper/leg-gps`.

3.2. Generality of the LEG family

The LEG family is useful only in so much as it is able to accurately approximate other GP kernels. Here, we prove that in fact the LEG family is general: *any* stationary Lebesgue-integrable stationary continuous kernel can be approximated to arbitrary accuracy with a LEG family of a certain rank ℓ.

Intuitively the argument is as follows: first, the PEG family provides a general and well-behaved (stable, stationary, correlated) collection of ℓ state-space components. Second, the LEG observation model creates *mixture* of those underlying PEG components. Third, we show that the LEG family has nonzero intersection with spectral mixture kernels (a popular class of kernels defined more carefully below), thus drawing a novel and useful connection between spectral mixtures and state space models. Fourth, we extend known facts about the generality of spectral mixtures to the multidimensional case. As a result, finally, we conclude that the LEG family is general (all without sacrificing its linear

[1] Those familiar with the multigrid technique (Terzopoulos, 1986; Hackbusch, 2013) – which has been used for Gaussian inference in other contexts (Papandreou & Yuille, 2010; Mukadam et al., 2016; Zanella & Roberts, 2017) – will note similarities between multigrid and CR.

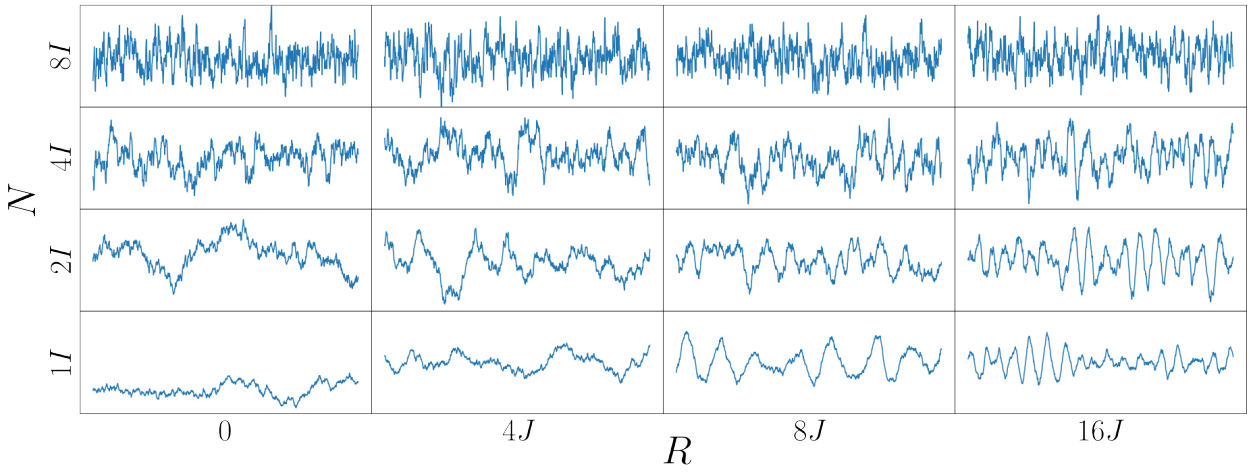

Figure 1. **PEG process samples**. The plots above show representative samples from the model $\mathrm{PEG}(N, R)$ as we vary N and R. Here we consider rank-2 PEG models (only one element of the 2d vector is plotted), so N, R are both 2×2 matrices. We vary N by taking it to be various multiples of the identity. We vary R by taking various multiples of J, the antisymmetric 2×2 matrix with zeros on the diagonal and ± 1 on the off-diagonal. In this simple rank-2 case, increasing N leads to a less predictable process and increasing R leads to faster oscillations.

runtime).

To begin, we study the spectral representation of the LEG kernel. If a kernel is stationary and continuous, Bochner's theorem guarantees it has a spectrum (Bhatia, 2015): a unique matrix-valued measure F such that

$$C(\tau) = \int e^{-i\tau\omega} dF(\omega).$$

Spectral Mixture (SM) methods offer a direct way to approximate any stationary kernel through its spectrum. For the purposes of this article we will define SM kernels as follows:

Definition 3. Let p denote a probability density on $\mathbb{R}$, let $b_1, b_2 \cdots b_\ell \in \mathbb{C}^n$, let $\mu \in \mathbb{R}^\ell$, and let $\gamma > 0$. The **Spectral Mixture** kernel, $C_{\mathrm{SM}}(t; p, b, \mu, \gamma)$, is given by

$$\sum_{k=1}^{\ell} \int e^{-i\xi x} b_k b_k^* \gamma p(\gamma(\xi - \mu_k)) d\xi.$$

We will say that C is **based on** p, since its spectrum is a sum of scaled and shifted versions of p.

Spectral Mixtures were first introduced in machine learning in (Wilson & Adams, 2013), where it was noted that any kernel which is the covariance of a weakly stationary mean square continuous random process $X : \mathbb{R} \to \mathbb{R}$ (or indeed $X : \mathbb{R}^n \to \mathbb{R}$) can be well-approximated using SM kernels. However, that result does not hold for our case, i.e. kernels for processes of the form $X : \mathbb{R} \to \mathbb{R}^n$. In this situation the spectrum of the kernel becomes a complex-matrix-valued measure (instead of an ordinary probability measure).

These mixture kernels have an interesting connection to LEG kernels: all Cauchy-based spectral mixture kernels are actually also LEG kernels. These kernels thus fall at the intriguing intersection of Gaussian Hidden Markov models (which are linear run-time) and Spectral Mixture models (which have not previously been considered linear run-time). Every Cauchy-based SM kernel can be understood as a LEG kernel. There is also another generalization of Cauchy-based SM kernels, known as the Celerite kernels (Foreman-Mackey et al., 2017); these are built by linear combinations of kernels called Celerite terms. Below we summarize the relationship between these three families of kernels:

Lemma 2 (SM kernels, Celerite kernels, LEG kernels)**.**

1. *Every Cauchy-based real-valued SM kernel* $C_{\mathrm{SM}} : \mathbb{R} \to \mathbb{R}$ *can be understood as a Celerite kernel.*
2. *Every positive-definite Celerite term can be understood as a LEG kernel.*
3. *Every Cauchy-based real-valued SM kernel* $C_{\mathrm{SM}} : \mathbb{R} \to \mathbb{R}^{n \times n}$ *can be understood as a LEG kernel.*

Proof: See supplementary material.

Thus, to prove that the family of LEG kernels is general, it suffices to show that SM kernels are general. The key idea is to generalize a classic result from kernel density estimation. We achieve this generalization in the following theorem:

Theorem 1 (Total variation convergence for weighted kernel density estimation)**.** *Let* K, p *denote bounded densities*

on $\mathbb{R}^d$. Let $g : \mathbb{R} \to [-M, M]$. Let $\gamma_\ell = \ell^{1/2d}$. Let $\mu_1, \mu_2 \cdots \sim p$, independently. For each $\ell \in 1, 2, \cdots$, define

$$h_\ell(\xi) = \frac{1}{\ell} \sum_{k=1}^{\ell} g(\mu_k)\gamma_\ell^d K(\gamma_\ell(\xi - \mu_k)).$$

Then

$$\mathbb{P}\left(\lim_{\ell\to\infty} \int |h_\ell(\xi) - p(\xi)g(\xi)|d\xi = 0\right) = 1.$$

Proof: See supplementary material.

With this theorem in place, we next show that spectral mixture kernels can approximate any integrable continuous kernel for a stationary Gaussian process on one dimension:

Corollary 1 (Flexibility of Spectral Mixture kernels)**.** *Fix p, a bounded probability density on $\mathbb{R}^n$, $\varepsilon > 0$, and any Lebesgue-integrable continuous positive definite[2] stationary kernel $\Sigma : \mathbb{R} \to \mathbb{C}^{n\times n}$. There exists a real valued kernel $C = C_{\mathrm{SM}}(p, b, \mu, \gamma)$ such that $\|C(\tau)z - \Sigma(\tau)z\| < \varepsilon\|z\|$ for every $\tau \in \mathbb{R}, z \in \mathbb{C}^n$.*

Proof: See supplementary material.

This corollary can be used to establish our main mathematical result, i.e., that LEG models enjoy the same flexibility guarantee.

Theorem 2 (Flexibility of LEG and Celerite kernels)**.** *For every $\varepsilon > 0$ and every Lebesgue-integrable continuous positive definite stationary kernel $\Sigma : \mathbb{R} \to \mathbb{R}^{n\times n}$ there exists a Celerite kernel C such that $\|C(\tau)z - \Sigma(\tau)z\| < \varepsilon\|z\|$ for every $\tau > 0, z \in \mathbb{C}^n$. Moreover, there exists a LEG kernel with the same guarantee.*

Proof: See supplementary material.

In conclusion, we have proven that any stationary Gaussian Process on one dimension can be well approximated using LEG processes, and further that the computational effort for LEG processes scales linearly with the number of observations. Thus, putting these two pieces together, approximate inference for any stationary Gaussian Processes on one dimension, at any desired level of accuracy, requires computational effort that scales linearly with the number of observations.

4. Extensions

Before moving on to illustrate these results with experiments on simulated and real data, we pause to note several useful extensions.

[2]The requirements of continuity and integrability are slightly too strong. For example, the sinc kernel is not Lebesgue integrable, but it is easy to approximate with a spectral mixture kernel. In the future we hope to refine these conditions.

4.1. Non-stationary processes

We have focused on stationary processes here for simplicity. A number of potential extensions to non-stationary processes are possible while retaining linear-time scaling. As one example, starting with LEG processes as a base, non-stationary models can be developed using the techniques from (Benavoli & Zaffalon, 2016).

4.2. Non-Gaussian observations

Many approaches have been developed to adapt GP inference methods to non-Gaussian observations, including Laplace approximations, expectation propagation, variational inference, and a variety of specialized Monte Carlo methods (Hartikainen et al., 2011; Riihimäki et al., 2014; Nguyen & Bonilla, 2014; Nishihara et al., 2014). Many of these can be easily adapted to the LEG model, using the fact that the sum of a block-tridiagonal matrix (from the precision matrix of the LEG prior evaluated at the sampled data points) plus a diagonal matrix (contributed by the likelihood term of each observed data point) is again block-tridiagonal, leading to linear-time updates (Smith & Brown, 2003; Paninski et al., 2010; Fahrmeir & Tutz, 2013; Polson et al., 2013; Khan & Lin, 2017; Nickisch et al., 2018).

4.3. Non-linear domains

Just as Gaussian Markov models in discrete time can be easily extended to Gaussian graphical models on general graphs, we can extend the Gaussian Markov PEG and LEG processes to stochastic processes on more general domains. In the simplest case the domain of the process could be a tree, with the PEG kernel defined in terms of distance along the tree, rather than distance on the line. Inference in the resulting tree-structured Gaussian graphical model can proceed via message passing in $O(m)$ time.

4.4. Multi-dimensional domains

We can also use LEG kernels to model processes of the form $x : \mathbb{R}^d \to \mathbb{R}^n$. Let B, Λ be matrices, let N, R be collections of matrices, and let $C_{\mathrm{KLEG}}(\tau; N, R, B, \Lambda) \triangleq \delta_\tau \Lambda\Lambda^\top + \sum_{r=1}^{\zeta} \prod_{k=1}^{d} BC_{\mathrm{PEG}}(\tau_k; N_{rk}, R_{rk})B^\top$.

Theorem 3. *Let $\Sigma : \mathbb{R}^d \to \mathbb{R}^{n\times n}$ any positive-definite integrable continuous stationary kernel, and fix $\varepsilon > 0$. There exists a KLEG kernel such that $\|C(\tau)z - C_{\mathrm{KLEG}}(\tau)z\| < \varepsilon z$.*

Proof: See supplementary material.

Efficient computation is possible for observations from a KLEG process along a (potentially irreguarly-spaced) grid. The covariance matrix of these observations has structure which can be leveraged for efficient computation. For example, in the supplement we give an algorithm for multiplying by this matrix, and show that the computational cost of this

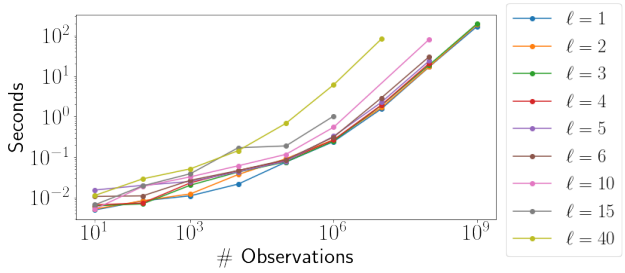

Figure 2. **Walltime for evaluating LEG likelihoods**. How long does it take to compute the likelihood of observations from a LEG model on an m5.24xlarge machine on the Amazon AWS service? We compare times for differently-ranked models and different numbers of observations. For example, the likelihood for one billion observations under a rank-3 LEG model can be computed in roughly three and a half minutes.

algorithm scales linearly with the number of points in the grid. Combined with GPyTorch (cf. (Gardner et al., 2018)), this algorithm should yield efficient inference algorithms for KLEG processes.

5. Experiments

Here we are interested in testing the theoretical results described above. In practice, how fast is inference with the LEG process? How well can the LEG model approximate popular kernels? How well does the LEG model extrapolate and interpolate? How well can it smooth?

5.1. Computational complexity

Throughout what follows, we will perform inference on LEG processes using the Cyclic Reduction algorithm outlined in 3.1. We wanted to check if there are practical difficulties that could negate the theoretically linear computational cost of this method. We measured how long it took to compute the likelihood of single contiguous chains of observations from LEG processes of various ranks. In each case we used an m5-24xlarge machine on Amazon Web Services (AWS).

Overall, the empirical scaling appeared consistent with the theoretical predictions. The likelihood of one million observations from a rank-3 model could be computed in 0.25 seconds, and one billion observations could be computed in 195 seconds. We saw similar trends across models of other ranks; the results are summarized in Figure 2. Note that for smaller datasets we actually observed a sublinear scaling (i.e. a slope of less than one on the log-log plot) that turns approximately linear for larger values of m.

5.2. Matching one-dimensional kernels

Theorem 2 shows that LEG kernels can represent any stationary Gaussian process arbitrarily well *if* the latent dimension ℓ is sufficiently high. How high does this dimension actually need to be in order to get a good fit? We investigate this question by examining several popular one-dimensional kernels. In each case we draw fifty thousand observations, each taken .1 units apart from the next. We fit LEG models of various ranks by optimizing the log likelihood using the BroydenFletcherGoldfarbShanno (BFGS) algorithm (as implemented in SciPy). Gradients were computed by TensorFlow2 using backpropogation through the Cyclic Reduction algorithm. We found this approach to be simple, scalable, and robust across datasets. The likelihood could also be optimized using Expectation Maximization, but we found it was not as fast (Dempster et al., 1977). The model could also by fit by moment-matching instead of likelihood; this is a common practice for ARMA models (Brockwell & Davis, 2013) and we hope to explore this possibility in the future.

How well do LEG kernels approximate the Rational Quadratic (RQ) kernel,

$$C_{\mathrm{RQ}}(\tau) = 2/(1+\tau^2)?$$

On the one hand, the Rational Quadratic kernel is profoundly different from every LEG kernel. The spectrum of the RQ kernel decays exponentially, whereas the spectrum of of a LEG process is asymptotically an inverse polynomial (recall from section 3.2 that PEG processes contain Cauchy spectral mixture models as a special case). On the other hand, Theorem 2 guarantees a Rational Quadratic kernel can be matched uniformly well by a LEG kernel. Figure 3 shows a rank-4 LEG kernel $C(\tau)$ appears to do an excellent job of matching the RQ kernel for $t < 2$ and an adequate job of matching for $t > 2$.

This apparent contradiction – RQ is profoundly different (on the tails of the spectrum) from every LEG kernel, yet every RQ kernel can be matched arbitrarily well by a LEG kernel – is resolved by considering the different timescales involved in any Gaussian Process. LEG kernels can uniformly approximate any stationary covariance, which means that we can use them to get uniformly accurate forecasts and interpolations at any fixed timescale. If a LEG kernel is trained on observations at a particular timescale, the kernel will attempt to match smoothness *at that timescale*. For example, these LEG kernels were trained on observations at a timescale of .1, so they will attempt to match the covariance at that scale and larger.

The case of the (RBF) kernel, given by $C_{\mathrm{RBF}}(\tau) = \exp(-\tau^2/2)$, is even more extreme. This kernel's spectrum decays log-quadratically – even faster than the spectrum of the RQ kernel. For any RBF process, any RQ process, and any PEG process, one can always find a small enough scale

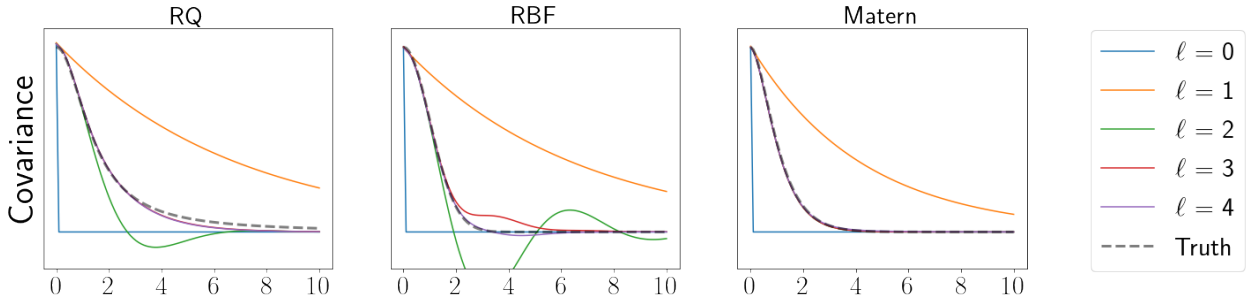

Figure 3. **LEG kernel approximation of some popular specific kernels**. By taking the rank ℓ sufficiently high we can achieve arbitrarily good approximations to any kernel. In the three examples shown here, $\ell = 4$ already provides adequate approximation quality. Note that in some cases some lines aren't visible because they are superimposed on each other; for example, when approximating Matern kernels we find nearly identical models for for $\ell \in 2, 3, 4$.

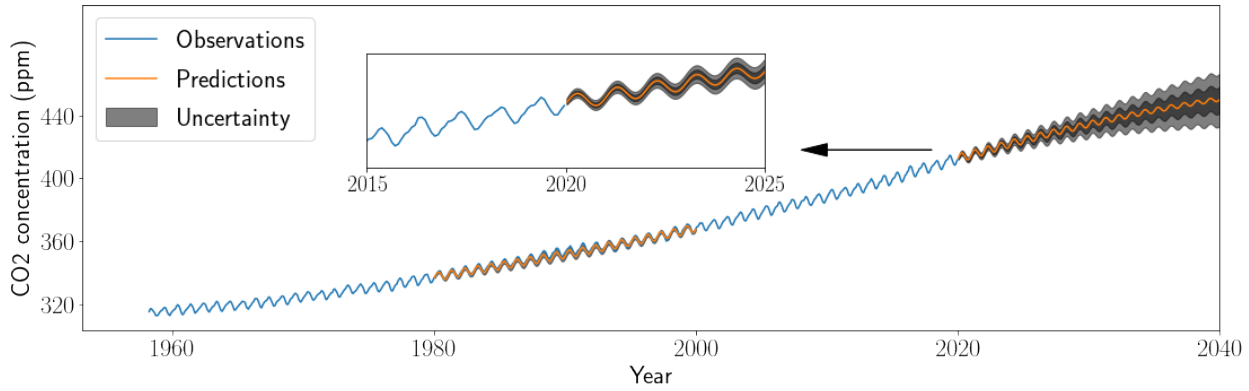

Figure 4. **LEG processes interpolate and extrapolate well across long timescales**. It appears that a rank-5 LEG model is sufficient to capture the linear and periodic trends in the Mauna Loa CO_2 dataset. Above we compare the true observations with interpolations made by the LEG model. The gray areas encompass one and two predictive standard deviations, i.e. the LEG model's uncertainty in forecasting and extrapolating what out-of-sample observations would look like.

so that the RBF process will look smoother than the RQ process and the RQ process is smoother than the PEG processes. Modeling this smoothness is difficult for lower-rank LEG models. For example, the best rank-2 LEG model includes a large oscillation that is not found in the ground-truth RBF kernel. It is not until rank 4 that the LEG process is able to match the kernel well.

Finally, the Matern kernel with $\nu = 1.5$ turns out to be an easy case. This kernel is given by

$$C_{\text{Matern}}(\tau) = (1 + \sqrt{3}\tau)\exp(-\sqrt{3}\tau).$$

Like PEG kernels, the spectrum of the Matern kernel decays slower than exponentially. In fact, this Matern kernel lies inside the rank-2 LEG family. Let

$$N = 3^{1/4}\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \qquad R = \sqrt{3}\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}.$$

Then the Matern kernel is given by $\text{LEG}(N, R, \mathbf{1}/\sqrt{2}, 0)$.

5.3. Mauna Loa CO_2

To find out whether LEG models can offer a practical tool for extrapolation and interpolation, we turn to the Mauna Loa CO_2 dataset. For the last sixty years, the monthly average atmosphere CO_2 concentrations at the the Mauna Loa Observatory in Hawaii have been recorded (Keeling & Whorf, 2005). This dataset is interesting because it features two different kinds of structures: an overall upward trend and a yearly cycle. To test the ability of the LEG model to learn these kinds of structures from data, we trained a rank-5 LEG kernel on all the data before 1980 and all the data after 2000. We then asked the LEG model to interpolate what happened in the middle and forecast what the concentration might look like in the next twenty years.

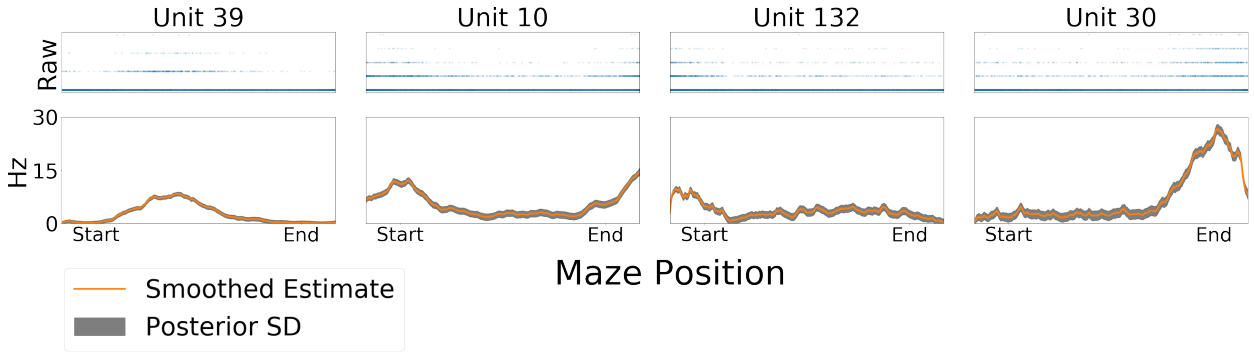

Figure 5. **LEG processes can smooth irregularly spaced neural data**. Ten thousand irregularly spaced observations suggest that the firing rates of hippocampal neurons are modulated by the rat's position in a maze. However, the modulation strength visible in the raw data is weak. By smoothing this data with a LEG process we can see the trend more clearly. How much should we smooth? By training a rank-5 LEG process we can determine a smoothness level automatically. The gray areas indicate the LEG model's posterior uncertainty about the estimated tuning curve.

The results are shown in Figure 4. It is encouraging that the LEG predictions interpolate adequately from 1980 to 1920. Even though the LEG process is given no exogenous information about "years" or "seasons," it correctly infers the number of number of bumps between 1980 and 1920. This example shows that the LEG model is sufficiently flexible to learn unanticipated structures in the data.

5.4. Hippocampal place-cells

Smoothing is another common application of GPs. Here we see whether LEG models can be used to smooth irregularly spaced observations from neural spiking data (Grosmark & Buzsáki, 2016).

In this data a rat's position in a one-dimensional maze is reported on a regular schedule, around 40 times per second. At each time-step, each neuron may be silent or may fire ("spike") some number of times. For each neuron, we would like to estimate the "tuning curve" – a function which takes in positions and returns the expected number of spikes as a function of the rat's position. With no smoothness assumptions on this function, the problem is impossible; the rat is never observed at exactly the same place twice. However, it is unclear how much smoothness should be assumed. Gaussian Processes offer a natural way to automatically learn an appropriate level of smoothness from the data itself. Note that the observed positions do not fall into a regularly spaced grid, so classical approaches such as the ARMA model cannot be applied.

Here we model this tuning curve using a PEG process, $z \sim \mathrm{PEG}(N, R)$. In this view, each data-point from the experiment constitutes a noisy observation of z. When the rat is at position t we model the distribution on the number of spikes observed in a small timebin as a Gaussian, with mean $Bz(t)$ and variance $\Lambda\Lambda^\top$. (It would be interesting to apply a non-Gaussian observation model here, as in, e.g., (Smith & Brown, 2003; Rahnama Rad & Paninski, 2010; Savin & Tkacik, 2016; Gao et al., 2016), and references therein; as noted in section 4, linear-time approximate inference is feasible in this setting and is an important direction for future work.)

For each neuron we train the parameters of a separate LEG model. We can then look at the posterior distribution on the underlying tuning curve z. The posterior mean of this process for various neurons is shown in Figure 5. We also represent one standard-deviation of the posterior variance of z with gray shading. Fitting the LEG model and looking at the posterior under the learned model appears to yield an effective Empirical Bayes approach for this kind of data.

6. Conclusion

We here make two advances in speeding up inference for Gaussian Processes on one dimension. First, we show that the LEG model, a particularly tractable continuous-time Gaussian hidden Markov process, can be used to approximate any GP with a stationary integrable continuous kernel, critically enabling linear runtime scaling in the number of observations. Second, we make this theoretical result practical by developing Cyclic Reduction-based algorithms to parallelize this computation, and sharing TensorFlow2-based implementations of these algorithms. We believe these advances will open up a wide variety of new applications for GP modeling in highly data-intensive areas involving data sampled at high rates and/or over long intervals, including geophysics, astronomy, high-frequency trading, molecular biology, neuroscience, and more.

Acknowledgements

Thanks to Jake Soloff for resolving a thorny point about matrix-valued measures.

References

Ambikasaran, S., Foreman-Mackey, D., Greengard, L., Hogg, D. W., and ONeil, M. Fast direct methods for gaussian processes. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):252–265, 2015.

Benavoli, A. and Zaffalon, M. State space representation of non-stationary gaussian processes. *arXiv preprint arXiv:1601.01544*, 2016.

Bhatia, R. *Positive Definite Matrices*. Princeton Series in Applied Mathematics. Princeton University Press, 2015. ISBN 9780691168258.

Brockwell, P. and Davis, R. *Time Series: Theory and Methods*. Springer Series in Statistics. Springer New York, 2013. ISBN 9781489900043.

Buesing, L., Macke, J. H., and Sahani, M. Learning stable, regularised latent models of neural population dynamics. *Network: Computation in Neural Systems*, 23(1-2):24–47, 2012.

Cheung, B. L. P., Riedner, B. A., Tononi, G., and Van Veen, B. D. Estimation of cortical connectivity from eeg using state-space models. *IEEE Transactions on Biomedical engineering*, 57(9):2122–2134, 2010.

Choudhary, N., Gillis, N., and Sharma, P. On approximating the nearest ω-stable matrix. *arXiv preprint arXiv:1901.03069*, 2019.

Coffey, W., Kalmykov, Y., and Waldron, J. *The Langevin Equation: With Applications to Stochastic Problems in Physics, Chemistry, and Electrical Engineering*. Series in contemporary chemical physics. World Scientific, 2004. ISBN 9789812384621.

Cunningham, J., Shenoy, K., and Sahani, M. Fast gaussian process methods for point process estimation. In *Proceedings of the International Conference on Machine Learning*, Proceedings of Machine Learning Research, 2008.

Cutajar, K., Osborne, M., Cunningham, J., and Filippone, M. Preconditioning kernel matrices. In *Proceedings of the International Conference on Machine Learning*, Proceedings of Machine Learning Research, 2016.

De G. Matthews, A. G., Van Der Wilk, M., Nickson, T., Fujii, K., Boukouvalas, A., León-Villagrá, P., Ghahramani, Z., and Hensman, J. Gpflow: A gaussian process library using tensorflow. *The Journal of Machine Learning Research*, 18(1):1299–1304, 2017.

De Jong, P. The likelihood for a state space model. *Biometrika*, 75(1):165–169, 1988.

Dempster, A. P., Laird, N. M., and Rubin, D. B. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

Eubank, R. and Wang, S. The equivalence between the cholesky decomposition and the kalman filter. *The American Statistician*, 56(1):39–43, 2002.

Fahrmeir, L. and Tutz, G. *Multivariate statistical modelling based on generalized linear models*. Springer Science & Business Media, 2013.

Foreman-Mackey, D., Agol, E., Ambikasaran, S., and Angus, R. Fast and scalable gaussian process modeling with applications to astronomical time series. *The Astronomical Journal*, 154(6):220, 2017.

Gao, Y., Archer, E. W., Paninski, L., and Cunningham, J. P. Linear dynamical neural population models through nonlinear embeddings. In *Advances in neural information processing systems*, pp. 163–171, 2016.

Gardner, J., Pleiss, G., Weinberger, K. Q., Bindel, D., and Wilson, A. G. Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. In *Advances in Neural Information Processing Systems*, pp. 7576–7586, 2018.

Gilboa, E., Saatçi, Y., and Cunningham, J. P. Scaling multidimensional inference for structured gaussian processes. *IEEE transactions on pattern analysis and machine intelligence*, 37(2):424–436, 2013.

Gonzalez, J., Low, Y., and Guestrin, C. Residual splash for optimally parallelizing belief propagation. In *Proceedings of Artificial Intelligence and Statistics*, 2009.

Grosmark, A. D. and Buzsáki, G. Diversity in neural firing dynamics supports both rigid and learned hippocampal sequences. *Science*, 351(6280):1440–1443, 2016.

Hackbusch, W. *Multi-Grid Methods and Applications*. Springer Series in Computational Mathematics. Springer Berlin Heidelberg, 2013. ISBN 9783662024270.

Hartikainen, J., Riihimäki, J., and Särkkä, S. Sparse spatiotemporal gaussian processes with general likelihoods. In *International Conference on Artificial Neural Networks*, pp. 193–200. Springer, 2011.

Hensman, J., Fusi, N., and Lawrence, N. D. Gaussian processes for big data. In *Uncertainty in Artificial Intelligence*, 2013.

Karvonen, T. and Sarkkä, S. Approximate state-space gaussian processes via spectral transformation. In *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing*, pp. 1–6. IEEE, 2016.

Keeling, C. D. and Whorf, T. Atmospheric carbon dioxide record from mauna loa. *Carbon Dioxide Research Group, Scripps Institution of Oceanography, University of California La Jolla, California*, pp. 92093–0444, 2005.

Khan, M. E. and Lin, W. Conjugate-computation variational inference. In *Proceedings of Artificial Intelligence and Statistics*, 2017.

Lindgren, F., Rue, H., and Lindström, J. An explicit link between gaussian fields and gaussian markov random fields: the stochastic partial differential equation approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(4):423–498, 2011.

Low, K. H., Yu, J., Chen, J., and Jaillet, P. Parallel gaussian process regression for big data: Low-rank representation meets markov approximation. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

Mergner, S. *Applications of State Space Models in Finance: An Empirical Analysis of the Time-varying Relationship Between Macroeconomics, Fundamentals and Pan-European Industry Portfolios*. Univ.-Verlag Göttingen, 2009. ISBN 9783941875227.

Mukadam, M., Yan, X., and Boots, B. Gaussian process motion planning. In *2016 IEEE international conference on robotics and automation*, pp. 9–15. IEEE, 2016.

Nguyen, T. V. and Bonilla, E. V. Automated variational inference for gaussian process models. In *Advances in Neural Information Processing Systems*, pp. 1404–1412, 2014.

Nickisch, H., Solin, A., and Grigorevskiy, A. State space Gaussian processes with non-Gaussian likelihood. In Dy, J. and Krause, A. (eds.), *Proceedings of the International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 3789–3798, Stockholmsmssan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

Nishihara, R., Murray, I., and Adams, R. P. Parallel mcmc with generalized elliptical slice sampling. *The Journal of Machine Learning Research*, 15(1):2087–2112, 2014.

Paninski, L., Ahmadian, Y., Ferreira, D. G., Koyama, S., Rad, K. R., Vidne, M., Vogelstein, J., and Wu, W. A new look at state-space models for neural data. *Journal of computational neuroscience*, 29(1-2):107–126, 2010.

Papandreou, G. and Yuille, A. L. Gaussian sampling by local perturbations. In *Advances in Neural Information Processing Systems*, pp. 1858–1866, 2010.

Polson, N. G., Scott, J. G., and Windle, J. Bayesian inference for logistic models using pólya–gamma latent variables. *Journal of the American statistical Association*, 108(504):1339–1349, 2013.

Quiñonero-Candela, J. and Rasmussen, C. E. A unifying view of sparse approximate gaussian process regression. *Journal of Machine Learning Research*, 6(Dec):1939–1959, 2005.

Rahnama Rad, K. and Paninski, L. Efficient, adaptive estimation of two-dimensional firing rate surfaces via gaussian process methods. *Network: Computation in Neural Systems*, 21(3-4):142–168, 2010.

Reinsel, G. *Elements of Multivariate Time Series Analysis*. Springer Series in Statistics. Springer New York, 2003. ISBN 9780387406190.

Riihimäki, J., Vehtari, A., et al. Laplace approximation for logistic gaussian process density estimation and regression. *Bayesian analysis*, 9(2):425–448, 2014.

Savin, C. and Tkacik, G. Estimating nonlinear neural response functions using gp priors and kronecker methods. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 29*, pp. 3603–3611. 2016.

Smith, A. C. and Brown, E. N. Estimating a state-space model from point process observations. *Neural computation*, 15(5):965–991, 2003.

Snelson, E. and Ghahramani, Z. Local and global sparse gaussian process approximations. In *Proceedings of Artificial Intelligence and Statistics*, 2007.

Sweet, R. A. A generalized cyclic reduction algorithm. *SIAM Journal on Numerical Analysis*, 11(3):506–520, 1974.

Terzopoulos, D. Image analysis using multigrid relaxation methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (2):129–139, 1986.

Vatiwutipong, P. and Phewchean, N. Alternative way to derive the distribution of the multivariate ornstein–uhlenbeck process. *Advances in Difference Equations*, 2019(1):1–7, 2019.

Wang, K., Pleiss, G., Gardner, J., Tyree, S., Weinberger, K. Q., and Wilson, A. G. Exact gaussian processes on a million data points. In *Advances in Neural Information Processing Systems*, pp. 14622–14632, 2019.

Wilson, A. and Adams, R. Gaussian process kernels for pattern discovery and extrapolation. In *Proceedings of the International Conference on Machine Learning*, pp. 1067–1075, 2013.

Wilson, A. and Nickisch, H. Kernel interpolation for scalable structured gaussian processes (kiss-gp). In *Proceedings of the International Conference on Machine Learning*, pp. 1775–1784, 2015.

Zanella, G. and Roberts, G. Analysis of the gibbs sampler for gaussian hierarchical models via multigrid decomposition. *arXiv preprint arXiv:1703.06098*, 2017.

Zhang, Y., Leithead, W. E., and Leith, D. J. Time-series gaussian process regression based on toeplitz computation of o (n 2) operations and o (n)-level storage. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pp. 3711–3716. IEEE, 2005.