
Fast Kernel Learning for Multidimensional Pattern Extrapolation

Andrew Gordon Wilson*
CMU

Elad Gilboa*
WUSTL

Arye Nehorai
WUSTL

John P. Cunningham
Columbia

Abstract

The ability to automatically discover patterns and perform extrapolation is an essential quality of intelligent systems. Kernel methods, such as Gaussian processes, have great potential for pattern extrapolation, since the kernel flexibly and interpretably controls the generalisation properties of these methods. However, automatically extrapolating large scale multidimensional patterns is in general difficult, and developing Gaussian process models for this purpose involves several challenges. A vast majority of kernels, and kernel learning methods, currently only succeed in smoothing and interpolation. This difficulty is compounded by the fact that Gaussian processes are typically only tractable for small datasets, and scaling an expressive kernel learning approach poses different challenges than scaling a standard Gaussian process model. One faces additional computational constraints, and the need to retain significant model structure for expressing the rich information available in a large dataset. In this paper, we propose a Gaussian process approach for large scale multidimensional pattern extrapolation. We recover sophisticated out of class kernels, perform texture extrapolation, inpainting, and video extrapolation, and long range forecasting of land surface temperatures, all on large multidimensional datasets, including a problem with 383,400 training points. The proposed method significantly outperforms alternative scalable and flexible Gaussian process methods, in speed and accuracy. Moreover, we show that a distinct combination of expressive kernels, a fully non-parametric representation, and scalable inference which exploits existing model structure, are critical for large scale multidimensional pattern extrapolation.

1 Introduction

Our ability to effortlessly extrapolate patterns is a hallmark of intelligent systems: even with large missing regions in our field of view, we can see patterns and textures, and we can visualise in our mind how they generalise across space. Indeed machine learning methods aim to automatically learn and generalise representations to new situations. Kernel methods, such as Gaussian processes (GPs), are popular machine learning approaches for non-linear regression and classification [1, 2, 3]. Flexibility is achieved through a kernel function, which implicitly represents an inner product of arbitrarily many basis functions. The kernel interpretably controls the smoothness and generalisation properties of a GP. A well chosen kernel leads to impressive empirical performances [2].

However, it is extremely difficult to perform large scale multidimensional pattern extrapolation with kernel methods. In this context, the ability to learn a representation of the data entirely depends on *learning* a kernel, which is a priori unknown. Moreover, kernel learning methods [4] are not typically intended for automatic pattern extrapolation; these methods often involve hand crafting combinations of Gaussian kernels (for smoothing and interpolation), for specific applications such as modelling low dimensional structure in high dimensional data. Without human intervention, the vast majority of existing GP models are unable to perform pattern discovery and extrapolation.

*Authors contributed equally.

While recent approaches such as [5] enable extrapolation on small one dimensional datasets, it is difficult to generalise these approaches for larger multidimensional situations. These difficulties arise because Gaussian processes are computationally intractable on large scale data, and while scalable approximate GP methods have been developed [6, 7, 8, 9, 10, 11, 12, 13], it is uncertain how to best scale expressive kernel learning approaches. Furthermore, the need for flexible kernel learning on large datasets is especially great, since such datasets often provide more information to automatically learn an appropriate statistical representation.

In this paper, we introduce GPatt, a flexible, non-parametric, and computationally tractable approach to kernel learning for multidimensional pattern extrapolation, with particular applicability to data with grid structure, such as images, video, and spatial-temporal statistics. Specifically:

- We extend fast Kronecker-based GP inference (e.g., [14, 15]) to account for non-grid data. Our experiments include data where more than 70% of the training data are not on a grid. Indeed most applications where one would want to exploit Kronecker structure involve missing and non-grid data – caused by, e.g., water, government boundaries, missing pixels and image artifacts. By adapting expressive spectral mixture kernels to the setting of multidimensional inputs and Kronecker structure, we achieve exact inference and learning costs of $\mathcal{O}(PN^{\frac{P+1}{P}})$ computations and $\mathcal{O}(PN^{\frac{2}{P}})$ storage, for N datapoints and P input dimensions, compared to the standard $\mathcal{O}(N^3)$ computations and $\mathcal{O}(N^2)$ storage associated with GPs.
- We show that i) spectral mixture kernels (adapted for Kronecker structure); ii) scalable inference based on Kronecker methods (adapted for incomplete grids); and, iii) truly non-parametric representations, when used *in combination* (to form GPatt) distinctly enable large-scale multidimensional pattern extrapolation with GPs. We demonstrate this through a comparison with various expressive models and inference techniques: i) spectral mixture kernels with arguably the most popular scalable GP inference method (FITC) [10]; ii) a flexible and efficient recent spectral based kernel learning method (SSGP) [6]; and, iii) the most popular GP kernels with Kronecker based inference.
- The information capacity of non-parametric methods grows with the size of the data. A truly non-parametric GP must have a kernel that is derived from an infinite basis function expansion. We find that a truly non-parametric representation is necessary for pattern extrapolation on large datasets, and provide insights into this surprising result.
- GPatt is highly scalable and accurate. This is the first time, as far as we are aware, that highly expressive *non-parametric* kernels with in some cases hundreds of hyperparameters, on datasets exceeding $N = 10^5$ training instances, can be learned from the marginal likelihood of a GP, in only minutes. Such experiments show that one can, to some extent, solve kernel selection, and automatically extract useful features from the data, on large datasets, using a special combination of expressive kernels and scalable inference.
- We show the proposed methodology provides a distinct approach to texture extrapolation and inpainting; it was not previously known how to make GPs work for these fundamental applications.
- Moreover, unlike typical inpainting approaches, such as patch-based methods (which work by recursively copying pixels or patches into a gap in an image, preserving neighbourhood similarities), GPatt is not restricted to spatial inpainting. This is demonstrated on a video extrapolation example, for which standard inpainting methods would be inapplicable [16]. Similarly, we apply GPatt to perform large-scale long range forecasting of land surface temperatures, through learning a sophisticated correlation structure across space and time. This learned correlation structure also provides insights into the underlying statistical properties of these data.
- We demonstrate that GPatt can precisely recover sophisticated out-of-class kernels automatically.

2 Spectral Mixture Product Kernels for Pattern Discovery

The spectral mixture kernel has recently been introduced [5] to offer a flexible kernel that can learn any stationary kernel. By appealing to Bochner’s theorem [17] and building a scale mixture of A Gaussian pairs in the spectral domain, [5] produced the spectral mixture kernel

$$k_{\text{SM}}(\tau) = \sum_{a=1}^A w_a^2 \exp\{-2\pi^2 \tau^2 \sigma_a^2\} \cos(2\pi\tau\mu_a), \quad (1)$$

which they applied to one-dimensional input data with a small number of points. For tractability with multidimensional inputs and large data, we propose a spectral mixture product (SMP) kernel:

$$k_{\text{SMP}}(\tau|\boldsymbol{\theta}) = \prod_{p=1}^P k_{\text{SM}}(\tau_p|\boldsymbol{\theta}_p), \quad (2)$$

where τ_p is the p^{th} component of $\tau = x - x' \in \mathbb{R}^P$, $\boldsymbol{\theta}_p$ are the hyperparameters $\{\mu_a, \sigma_a^2, w_a^2\}_{a=1}^A$ of the p^{th} spectral mixture kernel in the product of Eq. (2), and $\boldsymbol{\theta} = \{\boldsymbol{\theta}_p\}_{p=1}^P$ are the hyperparameters of the SMP kernel. The SMP kernel of Eq. (2) has Kronecker structure which we exploit for scalable and exact inference in section 2.1. With enough components A , the SMP kernel of Eq. (2) can model any stationary product kernel to arbitrary precision, and is flexible even with a small number of components, since scale-location Gaussian mixture models can approximate many spectral densities. We use SMP- A as shorthand for an SMP kernel with A components in each dimension (for a total of $3PA$ kernel hyperparameters and 1 noise hyperparameter). Wilson [18, 19] contains detailed discussions of spectral mixture kernels.

Critically, a GP with an SMP kernel is not a finite basis function method, but instead corresponds to a finite (A component) mixture of infinite basis function expansions. Therefore such a GP is a truly nonparametric method. This difference between a *truly nonparametric* representation – namely a mixture of infinite bases – and a parametric kernel method, a finite basis expansion corresponding to a *degenerate* GP, is critical both conceptually and practically, as our results will show.

2.1 Fast Exact Inference with Spectral Mixture Product Kernels

Gaussian process inference and learning requires evaluating $(K + \sigma^2 I)^{-1} \mathbf{y}$ and $\log |K + \sigma^2 I|$, for an $N \times N$ covariance matrix K , a vector of N datapoints \mathbf{y} , and noise variance σ^2 , as described in the supplementary material. For this purpose, it is standard practice to take the Cholesky decomposition of $(K + \sigma^2 I)$ which requires $\mathcal{O}(N^3)$ computations and $\mathcal{O}(N^2)$ storage, for a dataset of size N . However, many real world applications are engineered for grid structure, including spatial statistics, sensor arrays, image analysis, and time sampling. [14] has shown that the Kronecker structure in product kernels can be exploited for exact inference and hyperparameter learning in $\mathcal{O}(PN^{\frac{2}{P}})$ storage and $\mathcal{O}(PN^{\frac{P+1}{P}})$ operations, so long as the inputs $x \in \mathcal{X}$ are on a multidimensional grid, meaning $\mathcal{X} = \mathcal{X}_1 \times \dots \times \mathcal{X}_P \subset \mathbb{R}^P$. Details are in the supplement.

Here we relax this grid assumption. Assuming we have a dataset of M observations which are not necessarily on a grid, we propose to form a complete grid using W imaginary observations, $\mathbf{y}_W \sim \mathcal{N}(\mathbf{f}_W, \epsilon^{-1} I_W)$, $\epsilon \rightarrow 0$. The total observation vector $\mathbf{y} = [\mathbf{y}_M, \mathbf{y}_W]^\top$ has $N = M + W$ entries: $\mathbf{y} = \mathcal{N}(\mathbf{f}, D_N)$, where the noise covariance matrix $D_N = \text{diag}(D_M, \epsilon^{-1} I_W)$, $D_M = \sigma^2 I_M$. The imaginary observations \mathbf{y}_W have *no corrupting effect* on inference: the moments of the resulting predictive distribution are exactly the same as for the standard predictive distribution, namely $\lim_{\epsilon \rightarrow 0} (K_N + D_N)^{-1} \mathbf{y} = (K_M + D_M)^{-1} \mathbf{y}_M$ (proof in the supplement).

For inference, we must evaluate $(K_N + D_N)^{-1} \mathbf{y}$. Since D_N is not a scaled identity (as is the usual case in Kronecker methods), we cannot efficiently decompose $K_N + D_N$, but we can efficiently take matrix vector products involving K_N and D_N . We therefore use preconditioned conjugate gradients (PCG) [20] to compute $(K_N + D_N)^{-1} \mathbf{y}$, an iterative method involving only matrix vector products. We use the preconditioning matrix $C = D_N^{-1/2}$ to solve $C^\top (K_N + D_N) C \mathbf{z} = C^\top \mathbf{y}$. The preconditioning matrix C speeds up convergence by ignoring the imaginary observations \mathbf{y}_W . Exploiting the fast multiplication of Kronecker matrices, PCG takes $\mathcal{O}(JPN^{\frac{P+1}{P}})$ total operations (where the number of iterations $J \ll N$) to compute $(K_N + D_N)^{-1} \mathbf{y}$ to convergence within machine precision (supplement). This procedure can also be used to handle heteroscedastic noise.

For learning (hyperparameter training) we must evaluate the marginal likelihood (supplement). We cannot efficiently compute the $\log |K_M + D_M|$ complexity penalty in the marginal likelihood, because K_M is not a Kronecker matrix. We approximate the complexity penalty as

$$\log |K_M + D_M| = \sum_{i=1}^M \log(\lambda_i^M + \sigma^2) \approx \sum_{i=1}^M \log(\tilde{\lambda}_i^M + \sigma^2), \quad (3)$$

for noise variance σ^2 . We approximate the eigenvalues λ_i^M of K_M using the eigenvalues of K_N such that $\tilde{\lambda}_i^M = \frac{M}{N} \lambda_i^N$ for $i = 1, \dots, M$, which is particularly effective for large M (e.g. $M > 1000$)

[7]. [21] proves this eigenvalue approximation is asymptotically consistent (e.g., converges in the limit of large M), and [22] shows how one can bound the true eigenvalues by their approximation using PCA. Notably, only the log determinant (complexity penalty) term in the marginal likelihood undergoes a small approximation, and inference remains exact.

All remaining terms in the marginal likelihood can be computed exactly and efficiently using PCG. The total runtime cost of hyperparameter learning and exact inference with an incomplete grid is thus $\mathcal{O}(PN^{\frac{P+1}{P}})$. In image problems, for example, $P = 2$, and so the runtime complexity reduces to $\mathcal{O}(N^{1.5})$. Although the proposed inference can handle non-grid data, this inference is most suited to inputs where there is some grid structure – images, video, spatial statistics, etc. If there is no such grid structure (e.g., none of the training data fall onto a grid), then the computational expense necessary to augment the data with imaginary grid observations can be prohibitive. Although incomplete grids have been briefly considered in, e.g. [23], such approaches generally involve costly and numerically unstable rank 1 updates, inducing inputs, and separate (and restricted) treatments of ‘missing’ and ‘extra’ data. Moreover, the marginal likelihood, critical for kernel learning, is not typically considered in alternate approaches to incomplete grids.

3 Experiments

In our experiments we combine the SMP kernel of Eq. (2) with the fast exact inference and learning procedures of section 2.1, in a GP method we henceforth call GPatt^{1,2}.

We contrast GPatt with many alternative Gaussian process kernel methods. We are particularly interested in kernel methods, since they are considered to be general purpose regression methods, but conventionally have difficulty with large scale multidimensional pattern extrapolation. Specifically, we compare to the recent sparse spectrum Gaussian process regression (SSGP) [6] method, which provides fast and flexible kernel learning. SSGP models the kernel spectrum (spectral density) as a sum of point masses, such that SSGP is a finite basis function (*parametric*) model, with as many basis functions as there are spectral point masses. SSGP is similar to the recent models of Le et al. [8] and Rahimi and Recht [9], except it learns the locations of the point masses through marginal likelihood optimization. We use the SSGP implementation provided by the authors at <http://www.tsc.uc3m.es/~miguel/downloads.php>.

To further test the importance of the fast inference (section 2.1) used in GPatt, we compare to a GP which uses the SMP kernel of section 2 but with the popular fast FITC [10, 24] inference, which uses inducing inputs, and is implemented in GPML (<http://www.gaussianprocess.org/gpml>). We also compare to GPs with the popular squared exponential (SE), rational quadratic (RQ) and Matérn (MA) (with 3 degrees of freedom) kernels, catalogued in Rasmussen and Williams [1], respectively for smooth, multi-scale, and finitely differentiable functions. Since GPs with these kernels cannot scale to the large datasets we consider, we combine these kernels with the same fast inference techniques that we use with GPatt, to enable a comparison.³ Moreover, we stress test each of these methods in terms of speed and accuracy, as a function of available data and extrapolation range, and number of components. *All* of our experiments contain a large percentage of non-grid data, and we test accuracy and efficiency as a function of the percentage of missing data.

In all experiments we assume Gaussian noise, to express the marginal likelihood of the data $p(\mathbf{y}|\boldsymbol{\theta})$ solely as a function of kernel hyperparameters $\boldsymbol{\theta}$. To learn $\boldsymbol{\theta}$ we optimize the marginal likelihood using BFGS. We use a simple initialisation scheme: any frequencies $\{\mu_a\}$ are drawn from a uniform distribution from 0 to the Nyquist frequency (1/2 the sampling rate), length-scales $\{1/\sigma_a\}$ from a truncated Gaussian distribution, with mean proportional to the range of the data, and weights $\{w_a\}$ are initialised as the empirical standard deviation of the data divided by the number of components used in the model. In general, we find GPatt is robust to initialisation, particularly for $N > 10^4$ datapoints. We show a representative initialisation in the experiments.

This range of tests allows us to separately understand the effects of the SMP kernel, a non-parametric representation, and the proposed inference methods of section 2.1; we will show that all are required for good extrapolation performance.

¹We write *GPatt-A* when GPatt uses an SMP-A kernel.

²Experiments were run on a 64bit PC, with 8GB RAM and a 2.8 GHz Intel i7 processor.

³We also considered the model of [25], but this model is intractable for the datasets we considered and is not structured for the fast inference of section 2.1.

3.1 Extrapolating Metal Tread Plate and Pores Patterns

We extrapolate the missing region, shown in Figure 1a, on a real metal tread plate texture. There are 12675 training instances (Figure 1a), and 4225 test instances (Figure 1b). The inputs are pixel locations $x \in \mathbb{R}^2$ ($P = 2$), and the outputs are pixel intensities. The full pattern is shown in Figure 1c. This texture contains shadows and subtle irregularities, no two identical diagonal markings, and patterns that have correlations across both input dimensions.

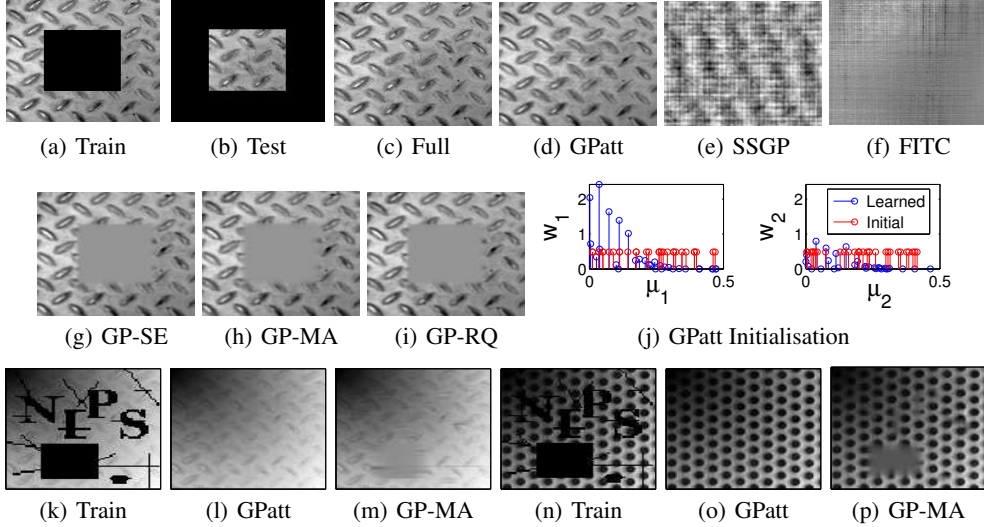


Figure 1: (a)-(j): Extrapolation on a Metal Tread Plate Pattern. Missing data are shown in black. a) Training region (12675 points), b) Testing region (4225 points), c) Full tread plate pattern, d) GPatt-30, e) SSGP with 500 basis functions, f) FITC with 500 inducing (pseudo) inputs, and the SMP-30 kernel, and GPs with the fast exact inference in section 2.1, and g) squared exponential (SE), h) Matérn (MA), and i) rational quadratic (RQ) kernels. j) Initial and learned hyperparameters using GPatt using simple initialisation. During training, weights of extraneous components automatically shrink to zero. (k)-(h) and (n)-(p): Extrapolation on tread plate and pore patterns, respectively, with added artifacts and non-stationary lighting changes.

To reconstruct the missing *and* training regions, we use GPatt-30. The GPatt reconstruction shown in Fig 1d is as plausible as the true full pattern shown in Fig 1c, and largely automatic. Without hand crafting of kernel features to suit this image, exposure to similar images, or a sophisticated initialisation, GPatt has automatically discovered the underlying structure of this image, and extrapolated that structure across a large missing region, even though the structure of this pattern is not independent across the two spatial input dimensions. Indeed the separability of the SMP kernel represents only a soft prior assumption, and does not rule out posterior correlations between input dimensions.

The reconstruction in Figure 1e was produced with SSGP, using 500 basis functions. In principle SSGP can model any spectral density (and thus any stationary kernel) with infinitely many components (basis functions). However, since these components are point masses (in frequency space), each component has highly limited expressive power. Moreover, with many components SSGP experiences practical difficulties regarding initialisation, over-fitting, and computation time (scaling quadratically with the number of basis functions). Although SSGP does discover some interesting structure (a diagonal pattern), and has equal training and test performance, it is unable to capture enough information for a convincing reconstruction, and we did not find that more basis functions improved performance. Likewise, FITC with an SMP-30 kernel and 500 inducing (pseudo) inputs cannot capture the necessary information to interpolate or extrapolate. On this example, FITC ran for 2 days, and SSGP-500 for 1 hour, compared to GPatt which took under 5 minutes.

GPs with SE, MA, and RQ kernels are all truly Bayesian nonparametric models – these kernels are derived from infinite basis function expansions. Therefore, as seen in Figure 1 g), h), i), these methods are completely able to capture the information in the training region; however, these kernels do not have the proper structure to reasonably extrapolate across the missing region – they simply act as smoothing filters. Moreover, this comparison is only possible because we have implemented these GPs using the fast exact inference techniques introduced in section 2.1.

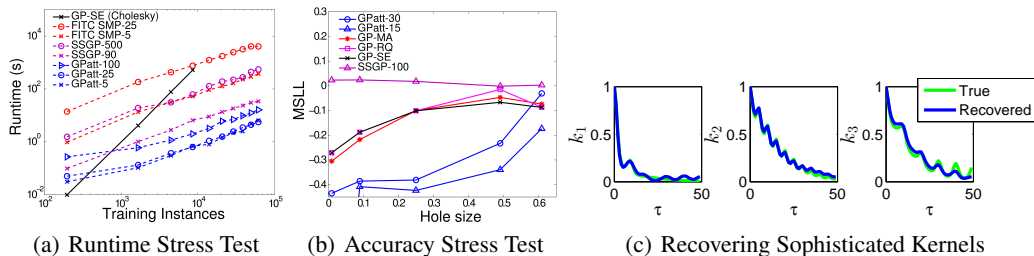


Figure 2: Stress Tests. a) **Runtime Stress Test.** We show the runtimes in seconds, as a function of training instances, for evaluating the log marginal likelihood, and any relevant derivatives, for a standard GP with SE kernel (as implemented in GPML), FITC with 500 inducing (pseudo) inputs and SMP-25 and SMP-5 kernels, SSGP with 90 and 500 basis functions, and GPatt-100, GPatt-25, and GPatt-5. Runtimes are for a 64bit PC, with 8GB RAM and a 2.8 GHz Intel i7 processor, on the cone pattern ($P = 2$), shown in the supplement. The ratio of training inputs to the sum of imaginary and training inputs for GPatt is 0.4 and 0.6 for the smallest two training sizes, and 0.7 for all other training sets. b) **Accuracy Stress Test.** MSLL as a function of holesize on the metal pattern of Figure 1. The values on the horizontal axis represent the fraction of missing (testing) data from the full pattern (for comparison Fig 1a has 25% missing data). We compare GPatt-30 and GPatt-15 with GPs with SE, MA, and RQ kernels (and the inference of section 2.1), and SSGP with 100 basis functions. The MSLL for GPatt-15 at a holesize of 0.01 is -1.5886 . c) **Recovering Sophisticated Kernels.** A product of three kernels (shown in green) was used to generate a movie of 112,500 training points. From this data, GPatt-20 reconstructs these component kernels (the learned SMP-20 kernel is shown in blue). All kernels are a function of $\tau = x - x'$ and have been scaled by $k(0)$.

Overall, these results indicate that both expressive nonparametric kernels, such as the SMP kernel, and the specific fast inference in section 2.1, are needed to extrapolate patterns in these images.

We note that the SMP-30 kernel used with GPatt has more components than needed for this problem. However, as shown in Fig. 1j, if the model is overspecified, the complexity penalty in the marginal likelihood shrinks the weights ($\{w_a\}$ in Eq. (1)) of extraneous components, as a proxy for model selection – an effect similar to *automatic relevance determination* [26]. Components which do not significantly contribute to model fit are automatically pruned, as shrinking the weights decreases the eigenvalues of K and thus minimizes the complexity penalty (a sum of log eigenvalues). The simple GPatt initialization in Fig 1j is used in all experiments and is especially effective for $N > 10^4$.






In Figure 1 (k)-(h) and (n)-(p) we use GPatt to extrapolate on treadplate and pore patterns with added artifacts and lighting changes. GPatt still provides a convincing extrapolation – able to uncover both local and global structure. Alternative GPs with the inference of section 2.1 can interpolate small artifacts quite accurately, but have trouble with larger missing regions.

3.2 Stress Tests and Recovering Complex 3D Kernels from Video

We stress test GPatt and alternative methods in terms of speed and accuracy, with varying data-sizes, extrapolation ranges, basis functions, inducing (pseudo) inputs, and components. We assess accuracy using standardised mean square error (SMSE) and mean standardized log loss (MSLL) (a scaled negative log likelihood), as defined in Rasmussen and Williams [1] on page 23. Using the empirical mean and variance to fit the data would give an SMSE and MSLL of 1 and 0 respectively. Smaller SMSE and more negative MSLL values correspond to better fits of the data.

The runtime stress test in Figure 2a shows that the number of components used in GPatt does not significantly affect runtime, and that GPatt is much faster than FITC (using 500 inducing inputs) and SSGP (using 90 or 500 basis functions), even with 100 components (601 kernel hyperparameters). The slope of each curve roughly indicates the asymptotic scaling of each method. In this experiment, the standard GP (with SE kernel) has a slope of 2.9 , which is close to the cubic scaling we expect. All other curves have a slope of 1 ± 0.1 , indicating linear scaling with the number of training instances. However, FITC and SSGP are used here with a *fixed* number of inducing inputs and basis functions. More inducing inputs and basis functions should be used when there are more training instances – and these methods scale quadratically with inducing inputs and basis functions for a fixed number of training instances. GPatt, on the other hand, can scale linearly in runtime as a function of training

Table 1: We compare the test performance of GPatt-30 with SSGP (using 100 basis functions), and GPs using SE, MA, and RQ kernels, combined with the inference of section 3.2, on patterns with a train test split as in the metal treadplate pattern of Figure 1. We show the results as SMSE (MSLL).

train, test	 Rubber mat 12675, 4225	 Tread plate 12675, 4225	 Pores 12675, 4225	 Wood 14259, 4941	 Chain mail 14101, 4779
GPatt	0.31 (-0.57)	0.45 (-0.38)	0.0038 (-2.8)	0.015 (-1.4)	0.79 (-0.052)
SSGP	0.65 (-0.21)	1.06 (0.018)	1.04 (-0.024)	0.19 (-0.80)	1.1 (0.036)
SE	0.97 (0.14)	0.90 (-0.10)	0.89 (-0.21)	0.64 (1.6)	1.1 (1.6)
MA	0.86 (-0.069)	0.88 (-0.10)	0.88 (-0.24)	0.43 (1.6)	0.99 (0.26)
RQ	0.89 (0.039)	0.90 (-0.10)	0.88 (-0.048)	0.077 (0.77)	0.97 (-0.0025)

size, without any deterioration in performance. Furthermore, the fixed 2-3 orders of magnitude GPatt outperforms the alternatives is as practically important as asymptotic scaling.

The accuracy stress test in Figure 2b shows extrapolation (MSLL) performance on the metal tread plate pattern of Figure 1c with varying holesizes, running from 0% to 60% missing data for testing (for comparison the hole in Fig 1a has 25% missing data). GPs with SE, RQ, and MA kernels (and the fast inference of section 2.1) all steadily increase in error as a function of holesize. Conversely, SSGP does not increase in error as a function of holesize – with finite basis functions SSGP cannot extract as much information from larger datasets as the alternatives. GPatt performs well relative to the other methods, even with a small number of components. GPatt is particularly able to exploit the extra information in additional training instances: only when the holesize is so large that over 60% of the data are missing does GPatt’s performance degrade to the same level as alternative methods.

In Table 1 we compare the test performance of GPatt with SSGP, and GPs using SE, MA, and RQ kernels, for extrapolating five different patterns, with the same train test split as for the tread plate pattern in Figure 1. All patterns are shown in the supplement. GPatt consistently has the lowest SMSE and MSLL. Note that many of these datasets are sophisticated patterns, containing intricate details which are not strictly periodic, such as lighting irregularities, metal impurities, etc. Indeed SSGP has a periodic kernel (unlike the SMP kernel which is not strictly periodic), and is capable of modelling multiple periodic components, but does not perform as well as GPatt on these examples.

We also consider a particularly large example, where we use GPatt-10 to perform learning and exact inference on the *Pores* pattern, with 383,400 training points, to extrapolate a large missing region with 96,600 test points. The SMSE is 0.077, and the total runtime was 2800 seconds. Images of the successful extrapolation are shown in the supplement.

We end this section by showing that GPatt can accurately recover a wide range of kernels, even using a small number of components. To test GPatt’s ability to recover ground truth kernels, we simulate a $50 \times 50 \times 50$ movie of data (e.g. two spatial input dimensions, one temporal) using a GP with kernel $k = k_1 k_2 k_3$ (each component kernel in this product operates on a different input dimension), where $k_1 = k_{SE} + k_{SE} \times k_{PER}$, $k_2 = k_{MA} \times k_{PER} + k_{MA} \times k_{PER}$, and $k_3 = (k_{RQ} + k_{PER}) \times k_{PER} + k_{SE}$. ($k_{PER}(\tau) = \exp[-2 \sin^2(\pi \tau \omega) / \ell^2]$, $\tau = x - x'$). We use 5 consecutive 50×50 slices for testing, leaving a large number $N = 112500$ of training points, providing much information to learn the true generating kernels. Moreover, GPatt-20 reconstructs these complex out of class kernels in under 10 minutes, as shown in Fig 2c. In the supplement, we show true and predicted frames from the movie.

3.3 Wallpaper and Scene Reconstruction and Long Range Temperature Forecasting

Although GPatt is a general purpose regression method, it can also be used for inpainting: image restoration, object removal, etc. We first consider a wallpaper image stained by a black apple mark, shown in Figure 3. To remove the stain, we apply a mask and then separate the image into its three channels (red, green, and blue), resulting in 15047 pixels in each channel for training. In each channel we ran GPatt using SMP-30. We then combined the results from each channel to restore the image without any stain, which is impressive given the subtleties in the pattern and lighting.

In our next example, we wish to reconstruct a natural scene obscured by a prominent rooftop, shown in the second row of Figure 3a). By applying a mask, and following the same procedure as for the stain, this time with 32269 pixels in each channel for training, GPatt reconstructs the scene without the rooftop. This reconstruction captures subtle details, such as waves, with only a single

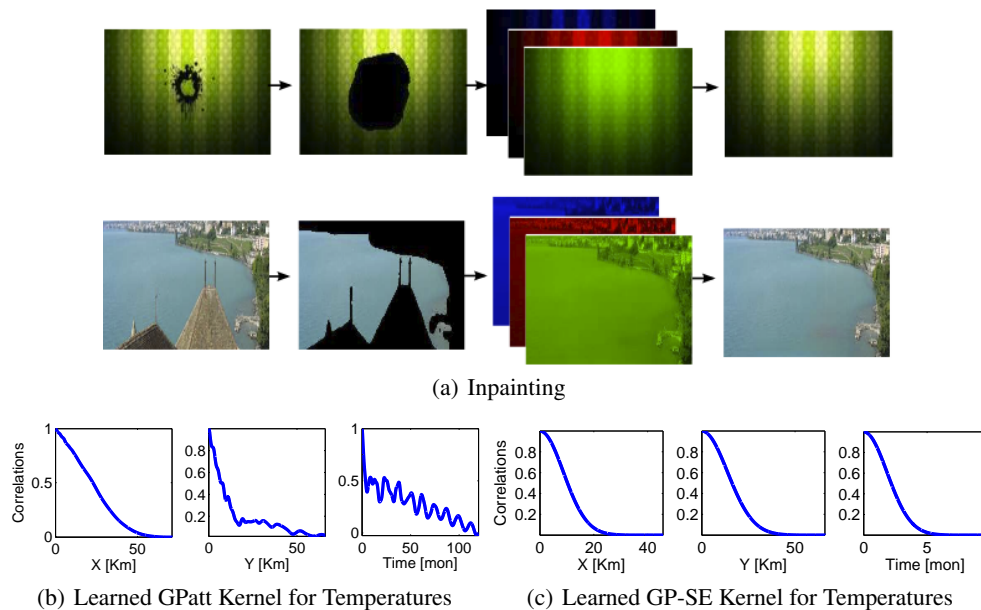


Figure 3: a) Image inpainting with GPatt. From left to right: A mask is applied to the original image, GPatt extrapolates the mask region in each of the three (red, blue, green) image channels, and the results are joined to produce the restored image. Top row: Removing a stain (train: 15047×3). Bottom row: Removing a rooftop to restore a natural scene (train: 32269×3). We do not extrapolate the coast. (b)-(c): Kernels learned for land surface temperatures using GPatt and GP-SE.

training image. In fact this example has been used with inpainting algorithms which were given access to a repository of thousands of similar images [27]. The results emphasized that conventional inpainting algorithms and GPatt have profoundly different objectives, which are sometimes even at cross purposes: inpainting attempts to make the image look good to a human (e.g., the example in [27] placed boats in the water), while GPatt is a general purpose regression algorithm, which simply aims to make accurate predictions at test input locations, from training data alone. For example, GPatt can naturally learn temporal correlations to make predictions in the video example of section 3.2, for which standard patch based inpainting methods would be inapplicable [16].

Similarly, we use GPatt to perform long range forecasting of land surface temperatures. After training on 108 months (9 years) of temperature data across North America (299,268 training points; a $71 \times 66 \times 108$ completed grid, with missing data for water), we forecast 12 months (1 year) ahead (33,252 testing points). The runtime was under 30 minutes. The learned kernels using GPatt and GP-SE are shown in Figure 3 b) and c). The learned kernels for GPatt are highly non-standard – both quasi periodic and heavy tailed. These learned correlation patterns provide insights into features (such as seasonal influences) which affect how temperatures vary in space and time. Indeed learning the kernel allows us to discover fundamental properties of the data. The temperature forecasts using GPatt and GP-SE, superimposed on maps of North America, are shown in the supplement.

4 Discussion

Large scale multidimensional pattern extrapolation problems are of fundamental importance in machine learning, where we wish to develop scalable models which can make impressive generalisations. However, there are many obstacles towards applying popular kernel methods, such as Gaussian processes, to these fundamental problems. We have shown that a combination of expressive kernels, truly Bayesian nonparametric representations, and inference which exploits model structure, can distinctly enable a kernel approach to these problems. Moreover, there is much promise in further exploring Bayesian nonparametric kernel methods for large scale pattern extrapolation. Such methods can be extremely expressive, and expressive methods are most needed for large scale problems, which provide relatively more information for automatically learning a rich statistical representation of the data.

Acknowledgements AGW thanks ONR grant N000141410684 and NIH grant R01GM093156. JPC thanks Simons Foundation grants SCGB #325171, #325233, and the Grossman Center at Columbia.

References

- [1] C.E. Rasmussen and C.K.I. Williams. *Gaussian processes for Machine Learning*. The MIT Press, 2006.
- [2] C.E. Rasmussen. *Evaluation of Gaussian Processes and Other Methods for Non-linear Regression*. PhD thesis, University of Toronto, 1996.
- [3] A. O’Hagan. Curve fitting and optimal design for prediction. *Journal of the Royal Statistical Society, B* (40):1–42, 1978.
- [4] M. Gönen and E. Alpaydm. Multiple kernel learning algorithms. *Journal of Machine Learning Research*, 12:2211–2268, 2011.
- [5] A.G. Wilson and R.P. Adams. Gaussian process kernels for pattern discovery and extrapolation. *International Conference on Machine Learning*, 2013.
- [6] M. Lázaro-Gredilla, J. Quiñero-Candela, C.E. Rasmussen, and A.R. Figueiras-Vidal. Sparse spectrum Gaussian process regression. *Journal of Machine Learning Research*, 11:1865–1881, 2010.
- [7] C.K.I. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems*, pages 682–688. MIT Press, 2001.
- [8] Q. Le, T. Sarlos, and A. Smola. Fastfood-computing Hilbert space expansions in loglinear time. In *International Conference on Machine Learning*, pages 244–252, 2013.
- [9] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Neural Information Processing Systems*, 2007.
- [10] E. Snelson and Z. Ghahramani. Sparse gaussian processes using pseudo-inputs. In *Advances in neural information processing systems*, volume 18, page 1257. MIT Press, 2006.
- [11] J. Hensman, N. Fusi, and N.D. Lawrence. Gaussian processes for big data. In *Uncertainty in Artificial Intelligence (UAI)*. AUAI Press, 2013.
- [12] M. Seeger, C.K.I. Williams, and N.D. Lawrence. Fast forward selection to speed up sparse Gaussian process regression. In *Workshop on AI and Statistics*, volume 9, 2003.
- [13] J. Quiñero-Candela and C.E. Rasmussen. A unifying view of sparse approximate Gaussian process regression. *The Journal of Machine Learning Research*, 6:1939–1959, 2005.
- [14] Y. Saatçi. *Scalable Inference for Structured Gaussian Process Models*. PhD thesis, University of Cambridge, 2011.
- [15] E. Gilboa, Y. Saatçi, and J.P. Cunningham. Scaling multidimensional inference for structured Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013.
- [16] C. Guillemot and O. Le Meur. Image inpainting: Overview and recent advances. *Signal Processing Magazine, IEEE*, 31(1):127–144, 2014.
- [17] S. Bochner. *Lectures on Fourier Integrals*, volume 42. Princeton University Press, 1959.
- [18] A.G. Wilson. A process over all stationary kernels. June, 2012. Technical Report, University of Cambridge. <http://www.cs.cmu.edu/~andrewgw/spectralkernel.pdf>.
- [19] A.G. Wilson. *Covariance Kernels for Fast Automatic Pattern Discovery and Extrapolation with Gaussian Processes*. PhD thesis, University of Cambridge, 2014. URL <http://www.cs.cmu.edu/~andrewgw/andrewgwthesis.pdf>.
- [20] K.E. Atkinson. *An introduction to numerical analysis*. John Wiley & Sons, 2008.
- [21] C.T.H. Baker. *The numerical treatment of integral equations*. 1977.
- [22] C.K.I. Williams and J. Shawe-Taylor. The stability of kernel principal components analysis and its relation to the process eigenspectrum. In *Advances in Neural Information Processing Systems*, volume 15, page 383. MIT Press, 2003.
- [23] Y. Luo and R. Duraiswami. Fast near-grid Gaussian process regression. In *International Conference on Artificial Intelligence and Statistics*, 2013.
- [24] A. Naish-Guzman and S. Holden. The generalized FITC approximation. In *Advances in Neural Information Processing Systems*, pages 1057–1064, 2007.
- [25] D. Duvenaud, J.R. Lloyd, R. Grosse, J.B. Tenenbaum, and Z. Ghahramani. Structure discovery in non-parametric regression through compositional kernel search. In *International Conference on Machine Learning*, 2013.
- [26] D.J.C MacKay. Bayesian nonlinear modeling for the prediction competition. *Ashrae Transactions*, 100 (2):1053–1062, 1994.
- [27] J. Hays and A. Efros. Scene completion using millions of photographs. *Communications of the ACM*, 51 (10):87–94, 2008.