Supplementary Material: Fast Kernel Learning for Multidimensional Pattern Extrapolation

Andrew Gordon Wilson* CMU
Elad Gilboa* WUSTL
Arye Nehorai WUSTL
John P. Cunningham Columbia

1 Introduction

We begin with background on Gaussian processes. We provide further detail about the eigendecomposition of kronecker matrices, and the runtime complexity of kronecker matrix vector products. We then provide images of the temperature forecasts. We also provide spectral images of the learned kernels in the metal tread plate experiment, larger versions of the images in Table 1, images of the extrapolation results on the large pore example, and images of the GPatt reconstruction for several consecutive movie frames. We also enlarge some of the results in the main text.

2 Gaussian Processes

A Gaussian process (GP) is a collection of random variables, any finite number of which have a joint Gaussian distribution. Using a Gaussian process, we can define a distribution over functions $f(x)$,

$$f(x) \sim \mathcal{GP}(m(x), k(x, x')) \,. \tag{1}$$

The mean function $m(x)$ and covariance kernel $k(x, x')$ are defined as

$$m(x) = \mathbb{E}[f(x)] \,, \tag{2}$$

$$k(x, x') = \text{cov}(f(x), f(x')) \,, \tag{3}$$

where x and x' are any pair of inputs in $\mathbb{R}^P$. Any collection of function values has a joint Gaussian distribution,

$$[f(x_1), \ldots, f(x_N)] \sim \mathcal{N}(\boldsymbol{\mu}, K) \,, \tag{4}$$

with mean vector $\boldsymbol{\mu}_i = m(x_i)$ and $N \times N$ covariance matrix $K_{ij} = k(x_i, x_j)$.

Assuming Gaussian noise, e.g. observations $y(x) = f(x) + \epsilon(x)$, $\epsilon(x) = \mathcal{N}(0, \sigma^2)$, the predictive distribution for $f(x_*)$ at a test input x_*, conditioned on $\mathbf{y} = (y(x_1), \ldots, y(x_N))^\top$ at training inputs $X = (x_1, \ldots, x_n)^\top$, is analytic and given by:

$$f(x_*)|x_*, X, \mathbf{y} \sim \mathcal{N}(\bar{f}_*, \mathrm{V}[f_*]) \tag{5}$$

$$\bar{f}_* = \mathbf{k}_*^\top (K + \sigma^2 I)^{-1} \mathbf{y} \tag{6}$$

$$\mathrm{V}[f_*] = k(x_*, x_*) - \mathbf{k}_*^\top (K + \sigma_n^2 I)^{-1} \mathbf{k}_* \,, \tag{7}$$

where the $N \times 1$ vector $\mathbf{k}_*$ has entries $(\mathbf{k}_*)_i = k(x_*, x_i)$.

The Gaussian process $f(x)$ can also be analytically marginalised to obtain the likelihood of the data, conditioned only on the hyperparameters $\boldsymbol{\theta}$ of the kernel:

$$\log p(\mathbf{y}|\boldsymbol{\theta}) \propto -[\overbrace{\mathbf{y}^\top (K_{\boldsymbol{\theta}} + \sigma^2 I)^{-1} \mathbf{y}}^{\text{model fit}} + \overbrace{\log |K_{\boldsymbol{\theta}} + \sigma^2 I|}^{\text{complexity penalty}}] \,. \tag{8}$$

*Authors contributed equally.

This *marginal likelihood* in Eq. (8) pleasingly compartmentalises into automatically calibrated model fit and complexity terms [1], and can be optimized to learn the kernel hyperparameters $\boldsymbol{\theta}$, or used to integrate out $\boldsymbol{\theta}$ using MCMC [2]. The problem of model selection and learning in Gaussian processes is "exactly the problem of finding suitable properties for the covariance function. Note that this gives us a model of the data, and characteristics (such as smoothness, length-scale, etc.) which we can interpret." [3].

The popular *squared exponential* (SE) kernel has the form

$$k_{\text{SE}}(x, x') = \exp(-0.5||x - x'||^2/\ell^2)\,. \tag{9}$$

GPs with SE kernels are smoothing devices, only able to learn how quickly sample functions vary with inputs x, through the length-scale parameter ℓ.

3 Eigendecomposition of Kronecker Matrices

Assuming a product kernel,

$$k(x_i, x_j) = \prod_{p=1}^{P} k^p(x_i^p, x_j^p)\,, \tag{10}$$

and inputs $x \in \mathcal{X}$ on a multidimensional grid, $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_P \subset \mathbb{R}^P$, then the covariance matrix K decomposes into a Kronecker product of matrices over each input dimension $K = K^1 \otimes \cdots \otimes K^P$ [4]. The eigendecomposition of K into $QVQ^\top$ similarly decomposes: $Q = Q^1 \otimes \cdots \otimes Q^P$ and $V = V^1 \otimes \cdots \otimes V^P$. Each covariance matrix K^p in the Kronecker product has entries $K^p_{ij} = k^p(x_i^p, x_j^p)$ and decomposes as $K^p = Q^p V^p {Q^p}^\top$. Thus the $N \times N$ covariance matrix K can be stored in $\mathcal{O}(PN^{\frac{2}{P}})$ and decomposed into $QVQ^\top$ in $\mathcal{O}(PN^{\frac{3}{P}})$ operations, for N datapoints and P input dimensions. [1] Moreover, the product of Kronecker matrices such as K, Q, or their inverses, with a vector $\mathbf{u}$, can be performed in $\mathcal{O}(PN^{\frac{P+1}{P}})$ operations (section 4).

Given the eigendecomposition of K as $QVQ^\top$, we can re-write $(K + \sigma^2 I)^{-1}\mathbf{y}$ and $\log|K + \sigma^2 I|$ (required for GP inference and learning as described in section 2)

$$(K + \sigma^2 I)^{-1}\mathbf{y} = (QVQ^\top + \sigma^2 I)^{-1}\mathbf{y} \tag{11}$$

$$= Q(V + \sigma^2 I)^{-1}Q^\top \mathbf{y}\,, \tag{12}$$

and

$$\log|K + \sigma^2 I| = \log|QVQ^\top + \sigma^2 I| = \sum_{i=1}^{N} \log(\lambda_i + \sigma^2)\,, \tag{13}$$

where λ_i are the eigenvalues of K, which can be computed in $\mathcal{O}(PN^{\frac{3}{P}})$.

Thus we can evaluate the predictive distribution and marginal likelihood to perform *exact* inference and hyperparameter learning, with $\mathcal{O}(PN^{\frac{2}{P}})$ storage and $\mathcal{O}(PN^{\frac{P+1}{P}})$ operations.

4 Matrix-vector Product for Kronecker Matrices

We first define a few operators from standard Kronecker literature. Let $\mathbf{B}$ be a matrix of size $p \times q$. The $\text{reshape}(\mathbf{B}, r, c)$ operator returns a r-by-c matrix ($rc = pq$) whose elements are taken column-wise from $\mathbf{B}$. The $\text{vec}(\cdot)$ operator stacks the matrix columns onto a single vector, $\text{vec}(\mathbf{B}) = \text{reshape}(\mathbf{B}, pq, 1)$, and the $\text{vec}^{-1}(\cdot)$ operator is defined as $\text{vec}^{-1}(\text{vec}(\mathbf{B})) = \mathbf{B}$. Finally, using the standard Kronecker property $(\mathbf{B} \otimes \mathbf{C})\text{vec}(\mathbf{X}) = \text{vec}(\mathbf{C}\mathbf{X}\mathbf{B}^\top)$, we note that for any N argument vector $\mathbf{u} \in \mathbb{R}^N$ we have

$$\mathbf{K}_N \mathbf{u} = \left(\bigotimes_{p=1}^{P} \mathbf{K}^p_{N^{1/P}}\right)\mathbf{u} = \text{vec}\left(\mathbf{K}^P_{N^{1/P}}\mathbf{U}\left(\bigotimes_{p=1}^{P-1} \mathbf{K}^p_{N^{1/P}}\right)^\top\right), \tag{14}$$

[1] The total number of datapoints $N = \prod_p |\mathcal{X}_p|$, where $|\mathcal{X}_p|$ is the cardinality of $\mathcal{X}_p$. For clarity of presentation, we assume each $|\mathcal{X}_p|$ has equal cardinality $N^{1/P}$.

where $\mathbf{U} = \text{reshape}(\mathbf{u}, N^{1/P}, N^{\frac{P-1}{P}})$, and $\mathbf{K}_N$ is an $N \times N$ Kronecker matrix. With no change to Eq. (14) we can introduce the $\text{vec}^{-1}(\text{vec}(\cdot))$ operators to get

$$\mathbf{K}_N \mathbf{u} = \text{vec}\left(\left(\begin{array}{c} \text{vec}^{-1}\left(\text{vec}\left(\begin{array}{c}\left(\bigotimes_{p=1}^{P-1} \mathbf{K}^p_{N^{1/P}}\right)\left(\mathbf{K}^P_{N^{1/P}}\mathbf{U}\right)^\top\end{array}\right)\right)\end{array}\right)^\top\right). \tag{15}$$

The inner component of Eq. (15) can be written as

$$\text{vec}\left(\left(\bigotimes_{p=1}^{P-1} \mathbf{K}^p_{N^{1/P}}\right)\left(\mathbf{K}^P_{N^{1/P}}\mathbf{U}\right)^\top \mathbf{I}_{N^{1/P}}\right) = \mathbf{I}_{N^{1/P}} \otimes \left(\bigotimes_{p=1}^{P-1} \mathbf{K}^p_{N^{1/P}}\right) \text{vec}\left(\left(\mathbf{K}^P_{N^{1/P}}\mathbf{U}\right)^\top\right). \tag{16}$$

Notice that Eq. (16) is in the same form as Eq. (14) (Kronecker matrix-vector product). By repeating Eqs. (15-16) over all P dimensions, and noting that $\left(\bigotimes_{p=1}^{P} \mathbf{I}_{N^{1/P}}\right)\mathbf{u} = \mathbf{u}$, we see that the original matrix-vector product can be written as

$$\left(\bigotimes_{p=1}^{P} \mathbf{K}^p_{N^{1/P}}\right)\mathbf{u} = \text{vec}\left(\left[\mathbf{K}^1_{N^{1/P}}, \ldots \left[\mathbf{K}^{P-1}_{N^{1/P}}, \left[\mathbf{K}^P_{N^{1/P}}, \mathbf{U}\right]\right]\right]\right) \tag{17}$$

$$\stackrel{\text{def}}{=} \text{kron\_mvprod}\left(\mathbf{K}^1_{N^{1/P}}, \mathbf{K}^2_{N^{1/P}}, \ldots, \mathbf{K}^P_{N^{1/P}}, \mathbf{u}\right) \tag{18}$$

where the bracket notation denotes matrix product, transpose then reshape, i.e.,

$$\left[\mathbf{K}^p_{N^{1/P}}, \mathbf{U}\right] = \text{reshape}\left(\left(\mathbf{K}^p_{N^{1/P}}\mathbf{U}\right)^\top, N^{1/P}, N^{\frac{P-1}{P}}\right). \tag{19}$$

Iteratively solving the kron_mvprod operator in Eq. (18) requires $(PN^{\frac{P+1}{P}})$, because each of the P bracket operations requires $\mathcal{O}(N^{\frac{P+1}{P}})$.

5 Inference with Missing Observations

In this paper, we extend Kronecker methods to account for training data which are not on a complete grid. Our approach, as described in the main text, involves augmenting the original training data with imaginary observations, in order to form a complete grid. Here, in the supplement, we show that the predictive distribution of a Gaussian process is unchanged by this procedure.

The predictive mean of a Gaussian process at L test points, given N training points, is given by

$$\boldsymbol{\mu}_L = \mathbf{K}_{LN}\left(\mathbf{K}_N + \sigma^2 I_N\right)^{-1}\mathbf{y}, \tag{20}$$

where $\mathbf{K}_{LN}$ is an $L \times N$ matrix of cross covariances between the test and training points. We wish to show that when we have M observations which are not on a grid that the desired predictive mean

$$\boldsymbol{\mu}_L = \mathbf{K}_{LM}\left(\mathbf{K}_M + \sigma^2 I_M\right)^{-1}\mathbf{y}_M = \mathbf{K}_{LN}\left(\mathbf{K}_N + \mathbf{D}_N\right)^{-1}\mathbf{y}, \tag{21}$$

where $\mathbf{y} = [\mathbf{y}_M, \mathbf{y}_W]^\top$ includes imaginary observations $\mathbf{y}_W$, and D_N is as defined in section 3 of the main paper as

$$D_N = \left[\begin{array}{cc} D_M & 0 \\ 0 & \epsilon^{-1} I_W \end{array}\right], \tag{22}$$

where we set $D_M = \sigma^2 I_M$.

Starting with the right hand side of Eq. (21),

$$\boldsymbol{\mu}_L = \left[\begin{array}{c} \mathbf{K}_{LM} \\ \mathbf{K}_{LW} \end{array}\right]\left[\begin{array}{cc} \mathbf{K}_M + \mathbf{D}_M & \mathbf{K}_{MW} \\ \mathbf{K}_{MW}^\top & \mathbf{K}_W + \epsilon^{-1}\mathbf{I}_W \end{array}\right]^{-1}\left[\begin{array}{c} \mathbf{y}_M \\ \mathbf{y}_W \end{array}\right]. \tag{23}$$

Using the block matrix inversion theorem, we get

$$\left[\begin{array}{cc} A & B \\ C & E \end{array}\right]^{-1} = \left[\begin{array}{cc} (A - BE^{-1}C)^{-1} & -A^{-1}B(I - E^{-1}CA^{-1}B)^{-1}E^{-1} \\ -E^{-1}C(A - BE^{-1}C)^{-1} & (I - E^{-1}CA^{-1}B)^{-1}E^{-1} \end{array}\right], \tag{24}$$

where $A = \mathbf{K}_M + \mathbf{D}_M$, $B = \mathbf{K}_{MW}$, $C = \mathbf{K}_{MW}^\top$, and $E = \mathbf{K}_W + \epsilon^{-1}\mathbf{I}_W$. If we take the limit of $E^{-1} = \epsilon(\epsilon\mathbf{K}_W + \mathbf{I}_W)^{-1} \overset{\epsilon \to 0}{\longrightarrow} \mathbf{0}$, and solve for the other components, Eq. (23) becomes

$$\boldsymbol{\mu}_L = \begin{bmatrix} \mathbf{K}_{LM} \\ \mathbf{K}_{LW} \end{bmatrix} \begin{bmatrix} (\mathbf{K}_M + \mathbf{D}_M)^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{y}_M \\ \mathbf{y}_W \end{bmatrix} = \mathbf{K}_{LM}(\mathbf{K}_M + \mathbf{D}_M)^{-1}\mathbf{y}_M \qquad (25)$$

which is the exact GP result. In other words, performing inference given observations $\mathbf{y}$ will give the same result as directly using observations $\mathbf{y}_M$. The proof that the predictive covariances remain unchanged proceeds similarly.

6 Land Temperature Forecasts

Figure 1 shows 12 month ahead forecasts for land surface temperatures using GPatt. We can see that GPatt has learned a representation of the training data and has made sensible long range extrapolations. The forecasts of GP-SE, with the popular squared exponential covariance function, quickly lose any relation with the training data.

7 Spectrum Analysis

We can gain further insight into the behavior of GPatt by looking at the spectral density learned by the spectral mixture kernel. Figure 2 shows the log spectrum representations of the learned kernels from Section 5.1. Smoothers, such as the popular kernels SE, RQ, and MA, concentrate their spectral energy around the origin, differing only by their support for higher frequencies. Methods which used the SMP kernel, such as the GPatt and FITC (with an SMP kernel), are able to learn meaningful features in the spectrum space.

8 Enlarged Inpainting Image

9 Tread Plate, Stress Test, and Video Images

Figure 4 illustrates the images used for the stress tests. In Figure 5, we provide the results for the large pore example. Finally, Figure 6 shows the true and predicted movie frames.

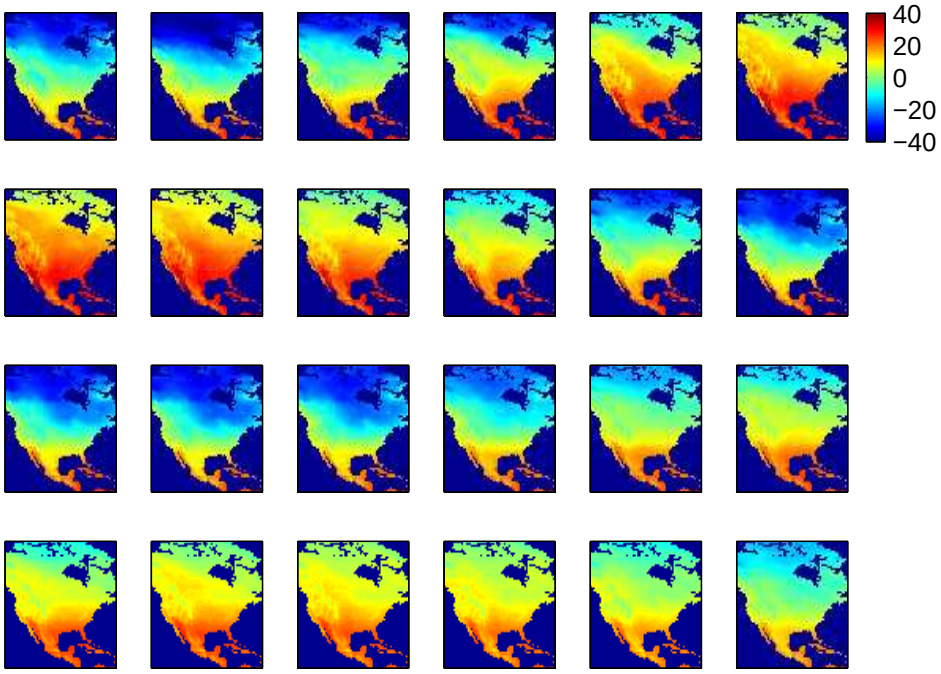
References

[1] C.E. Rasmussen and Z. Ghahramani. Occam's razor. In *Neural Information Process Systems*, 2001.

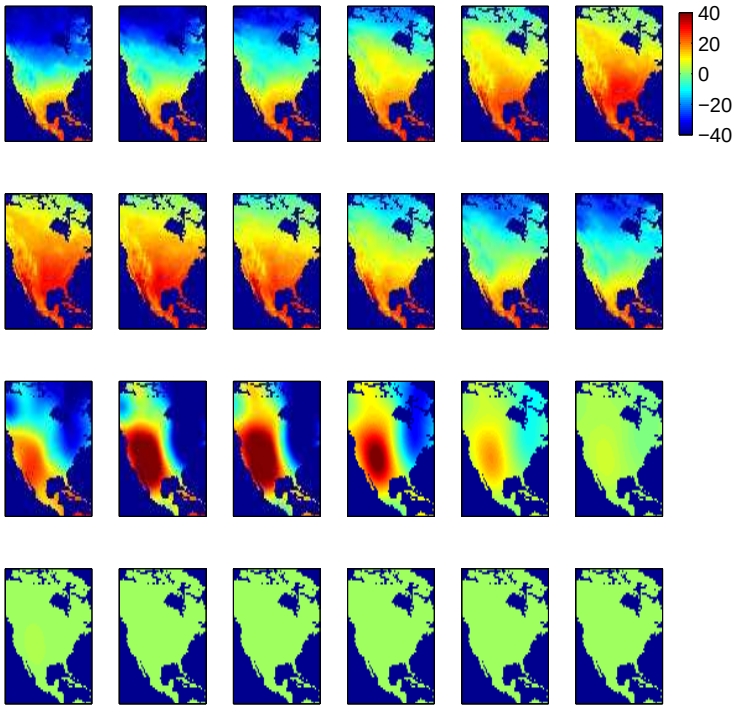
[2] I. Murray and R.P. Adams. Slice sampling covariance hyperparameters in latent Gaussian models. In *Advances in Neural Information Processing Systems*, 2010.

[3] C.E. Rasmussen and C.K.I. Williams. *Gaussian processes for Machine Learning*. The MIT Press, 2006.

[4] Y. Saatçi. *Scalable Inference for Structured Gaussian Process Models*. PhD thesis, University of Cambridge, 2011.

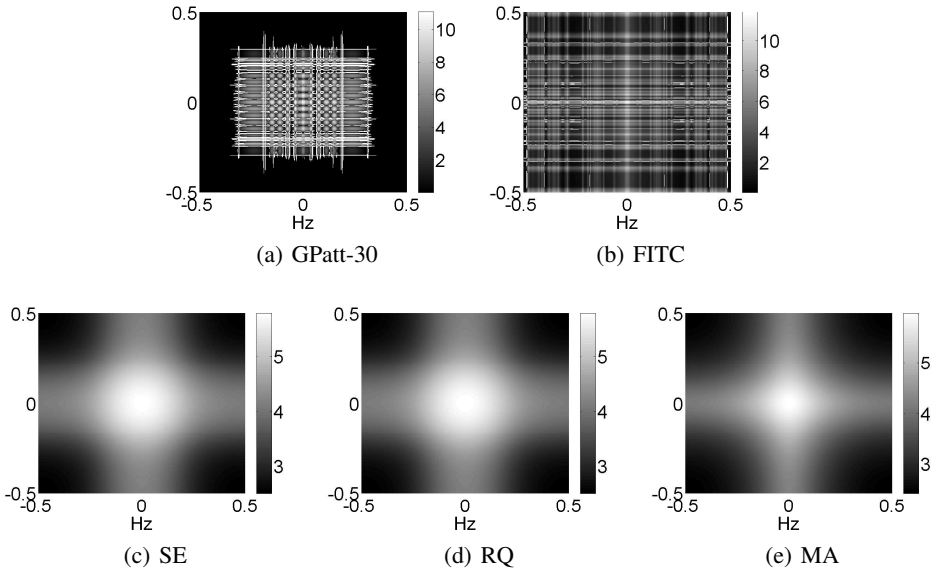


(a) GPatt



(b) GP-SE

Figure 1: In each image, the first two rows are the last 12 months of training data, and the last two rows are 12 month forecasts. Note that this is a true extrapolation problem: all 12 months are forecast at once (this is not a rolling forecast). a) GPatt, b) GP-SE.



(a) GPatt-30 (b) FITC

(c) SE (d) RQ (e) MA

Figure 2: Spectral representation of the learned kernels from Section 5.1. For methods which used the SMP kernel (namely, a) GPatt and b) FITC) we plot the analytical log spectrum using the learned hyperparameters. For c)Squared exponential, d) Rational quadratic, and e) Matérn-3 we plot instead the empirical log spectrum using the Fast Fourier transform of the kernel.

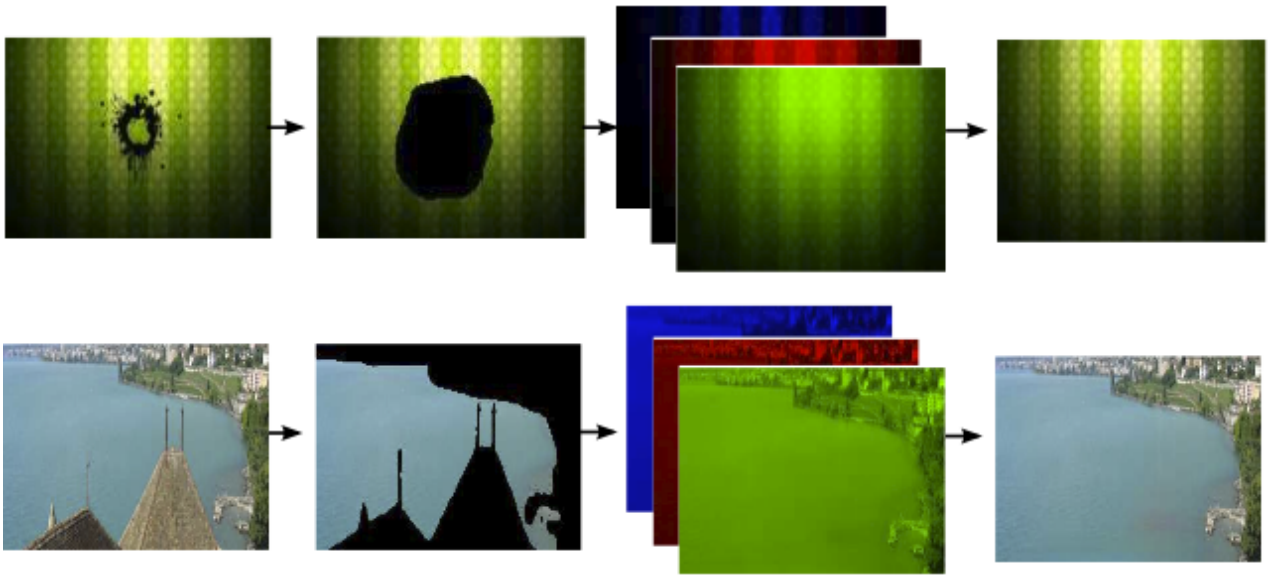
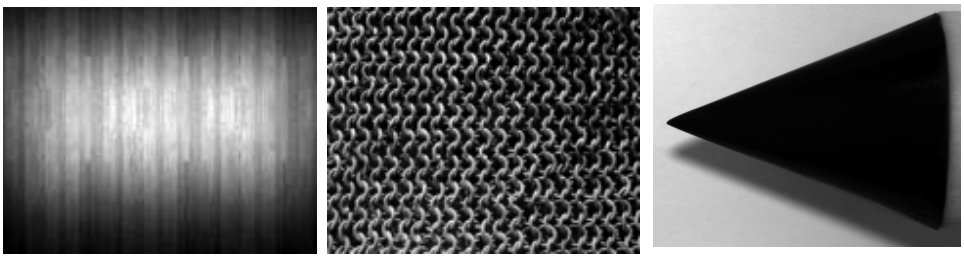


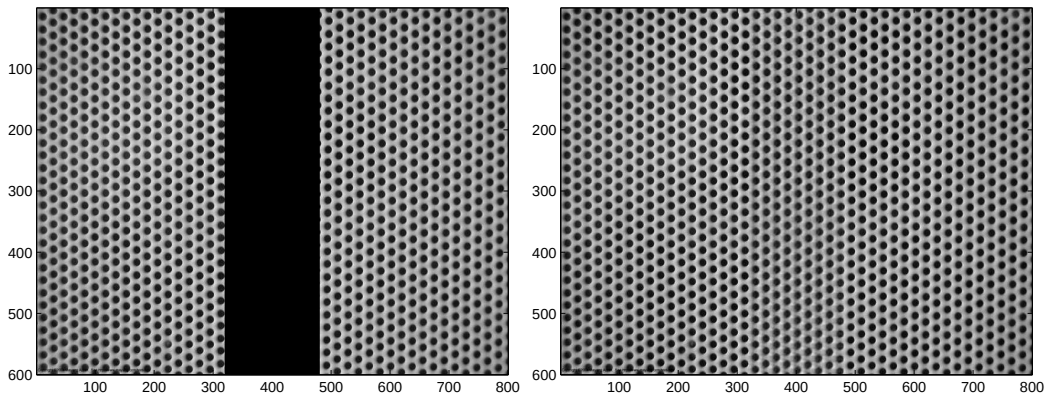
Figure 3: Image inpainting with GPatt. From left to right: A mask is applied to the original image, GPatt extrapolates the mask region in each of the three (red, blue, green) image channels, and the results are joined to produce the restored image. Top row: Removing a stain (train: 15047×3). Bottom row: Removing a rooftop to restore a natural scene (train: 32269×3). We do not attempt to extrapolate the coast, which is masked during training.

(a) Rubber mat (b) Tread plate (c) Pores



(d) Wood (e) Chain mail (f) Cone

Figure 4: Images used for stress tests in Section 5.2. Figures a) through e) show the textures used in the accuracy comparison of Table 1. Figure e) is the cone image which was used for the runtime analysis shown in Figure 3a.



(a) Train (b) Learn

Figure 5: GPatt on a particularly large multidimensional dataset. a) Training region (383400 points), b) GPatt-10 reconstruction of the missing region.

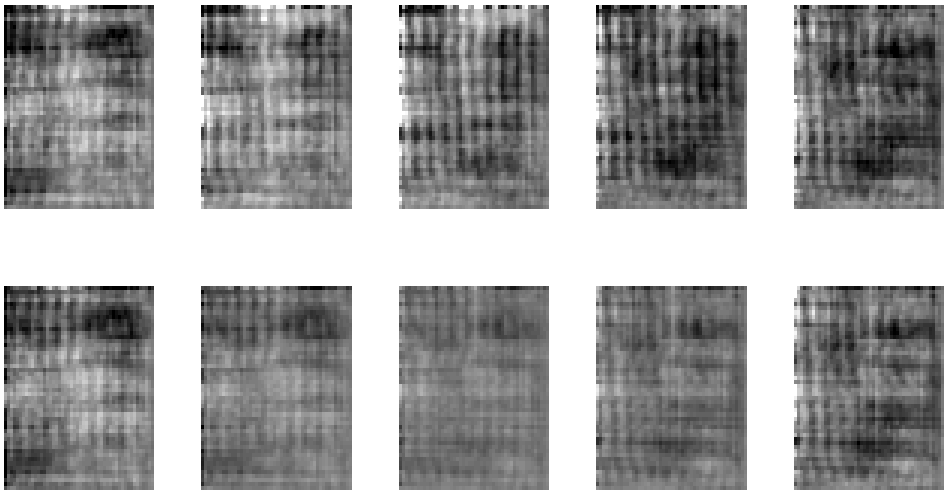


Figure 6: Recovering 5 consecutive slices from a movie. All 5 slices are missing during training: this is not one step ahead forecasting. (Top row: true slices take from the middle of the movie. Bottom row: predicted slices using GPatt-20.