Large-Scale Gaussian Processes for Spatiotemporal Modeling of Disease Incidence

Seth Flaxman

Machine Learning Department, Carnegie Mellon University
$\rightarrow$ Department of Statistics, Oxford

11 August 2015

Collaborators

- Flaxman, Wilson, Neill, Nickisch, and Smola. "Fast Kronecker Inference in Gaussian Processes with non-Gaussian Likelihoods," International Conference on Machine Learning 2015, Lille.
- Flaxman, Gelman, Neill, Smola, Vehtari, and Wilson, "Fast hierarchical Gaussian processes." [draft on my website]

Outline

- Large-scale spatiotemporal GP modeling
- Approximate and exact inference
- Hyperparameter learning
- Timing results on synthetic datasets
- Application: disease incidence
- Implementation

Gaussian process modeling

- Observations $y(\boldsymbol{s})$ at space/time locations $\boldsymbol{s}$

Gaussian process modeling

- Observations $y(\boldsymbol{s})$ at space/time locations $\boldsymbol{s}$
- Learn f such that $y(\boldsymbol{s}) = f(\boldsymbol{s}) + \epsilon$.

Gaussian process modeling

- Observations $y(\boldsymbol{s})$ at space/time locations $\boldsymbol{s}$
- Learn f such that $y(\boldsymbol{s}) = f(\boldsymbol{s}) + \epsilon$.
- GPs give a Bayesian framework for specifying a prior:

$$f(\boldsymbol{s}) \sim \mathcal{GP}(\mu(\boldsymbol{s}), k(\boldsymbol{s}, \boldsymbol{s}'))$$

Gaussian process modeling

- Observations $y(\boldsymbol{s})$ at space/time locations $\boldsymbol{s}$
- Learn f such that $y(\boldsymbol{s}) = f(\boldsymbol{s}) + \epsilon$.
- GPs give a Bayesian framework for specifying a prior:

$$f(\boldsymbol{s}) \sim \mathcal{GP}(\mu(\boldsymbol{s}), k(\boldsymbol{s}, \boldsymbol{s'}))$$

- Likelihood ("observation model"):

$$y(\boldsymbol{s})|f(\boldsymbol{s}) \sim \mathcal{N}(f(\boldsymbol{s}), \sigma^2 I)$$

Gaussian process modeling

- Observations $y(\mathbf{s})$ at space/time locations $\mathbf{s}$
- Learn f such that $y(\mathbf{s}) = f(\mathbf{s}) + \epsilon$.
- GPs give a Bayesian framework for specifying a prior:

$$f(\mathbf{s}) \sim \mathcal{GP}(\mu(\mathbf{s}), k(\mathbf{s}, \mathbf{s'}))$$

- Likelihood ("observation model"):

$$y(\mathbf{s})|f(\mathbf{s}) \sim \mathcal{N}(f(\mathbf{s}), \sigma^2 I)$$

- Likelihoods for count data:

$$y(\mathbf{s_i})|f(\mathbf{s_i}) \sim \text{Poisson}\left(\exp(f(\mathbf{s_i}))\right)$$
$$y(\mathbf{s_i})|f(\mathbf{s_i}) \sim \text{NegBinom}\left(\exp(f(\mathbf{s_i})), r\right)$$

Gaussian process modeling

- Observations $y(\mathbf{s})$ at space/time locations $\mathbf{s}$
- Learn f such that $y(\mathbf{s}) = f(\mathbf{s}) + \epsilon$.
- GPs give a Bayesian framework for specifying a prior:

$$f(\mathbf{s}) \sim \mathcal{GP}(\mu(\mathbf{s}), k(\mathbf{s}, \mathbf{s'}))$$

- Likelihood ("observation model"):

$$y(\mathbf{s})|f(\mathbf{s}) \sim \mathcal{N}(f(\mathbf{s}), \sigma^2 I)$$

- Likelihoods for count data:

$$y(\mathbf{s_i})|f(\mathbf{s_i}) \sim \text{Poisson}\left(\exp(f(\mathbf{s_i}))\right)$$
$$y(\mathbf{s_i})|f(\mathbf{s_i}) \sim \text{NegBinom}\left(\exp(f(\mathbf{s_i})), r\right)$$

- Combine prior and likelihood to get posterior

Why GPs for spatiotemporal data?

- Consistent non-parametric regression method [Choi & Schervish 2007, Van der Vaart and Van Zanten 2011]
- Rich structure in the mean function
- Flexible, expressive covariance functions
- Generalizes many spatial and time series models
- Inference can be as Bayesian as you like
- Much recent work on scaling up to large datasets
- Missing data and forecasting are automatic

Why GPs for spatiotemporal data?

- Consistent non-parametric regression method [Choi & Schervish 2007, Van der Vaart and Van Zanten 2011]
- Rich structure in the mean function
- Flexible, expressive covariance functions
- Generalizes many spatial and time series models
- Inference can be as Bayesian as you like
- Much recent work on scaling up to large datasets
- Missing data and forecasting are automatic

Expressive covariance functions

Spectral Mixture [Wilson & Adams 2013] kernel: scale-location mixture of $\mathcal{N}(\mu_q, v_q)$ in the spectral domain.
By Bochner's theorem, SM kernels can approximate any stationary covariance function.

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

w_q is the weight, $1/\mu_q$ is the period, and $1/\sqrt{v_q}$ is the length-scale.

Scaling up (the view from ML)

- Naively GP models are $\mathcal{O}(n^3)$ time complexity and $\mathcal{O}(n^2)$ space complexity
- Inducing points methods [see survey by Quiñonero-Candela and Rasmussen 2005]
- Variational inference [Titsias 2009, Hensman et al 2013]
- Kronecker methods: Bonilla et al. [2007], Finley et al. [2009], Stegle et al. [2011], Saati [2011], Gilboa et al. [2013], Riihimki and Vehtari [2014], Wilson et al. [2014], Groot et al. [2014]
- ...and many other ideas in spatial statistics!

Kronecker methods

Multivariate Gaussian distribution:

$$(2\pi)^{-n/2}|K|^{-1/2}e^{-\frac{1}{2}(x-\mu)^\top K^{-1}(x-\mu)}$$

Costly terms:

$$|K| \text{ and } K^{-1}$$

Assume observations on a grid and separable covariance:

$$K = K_s \otimes K_t$$

$$k((s,t),(s',t')) = k(s,s')k(t,t')$$

Then:

$$\det(K) = \prod_i \det(K_s)^m \det(K_t)^n$$

$$K^{-1}v = (K_s^{-1} \otimes K_t^{-1})v$$

Kronecker methods

Eigendecomposition $K_s = Q_1^\top \Lambda_1 Q_1$, $K_t = Q_2^\top \Lambda_2 Q_2$

$$K_s \otimes K_t = (Q_1^\top \otimes Q_2^\top)(\Lambda_1 \otimes \Lambda_2)(Q_1 \otimes Q_2)$$

$$K_s \otimes K_t + \sigma^2 I = (Q_1^\top \otimes Q_2^\top)(\Lambda_1 \otimes \Lambda_2 + \sigma^2 I)(Q_1 \otimes Q_2)$$

$$\log|K_s \otimes K_t + \sigma^2 I| = N_1 N_2 \sum_{ij} \log(\Lambda_{1ii}\Lambda_{2jj} + \sigma^2)$$

$$(K_s \otimes K_t + \sigma^2 I)^{-1} y =$$
$$\left((Q_1^\top \otimes Q_2^\top)(\Lambda_1 \otimes \Lambda_2 + \sigma^2 I)^{-1}(Q_1 \otimes Q_2)\right) y$$

Kronecker methods

- Runtime is nearly linear time: $\mathcal{O}(Dn^{\frac{D+1}{D}})$ for n observations and D dimensions.
- Memory requirements are negligible: $\mathcal{O}(Dn^{\frac{2}{D}}) \leq \mathcal{O}(n)$.
- Non-Gaussian observation models can be handled by the Laplace approximation (with an extra approximation for the log-determinant): Flaxman, Wilson, Neill, Nickisch, and Smola. "Fast Kronecker Inference in Gaussian Processes with non-Gaussian Likelihoods," ICML 2015.

Hyperparameter learning

- Back to our basic model:

$$f(\boldsymbol{s}) \sim \mathcal{GP}(\mu(\boldsymbol{s}), k_\theta(\boldsymbol{s}, \boldsymbol{s'}))$$

- How can we learn kernel hyperparameters?

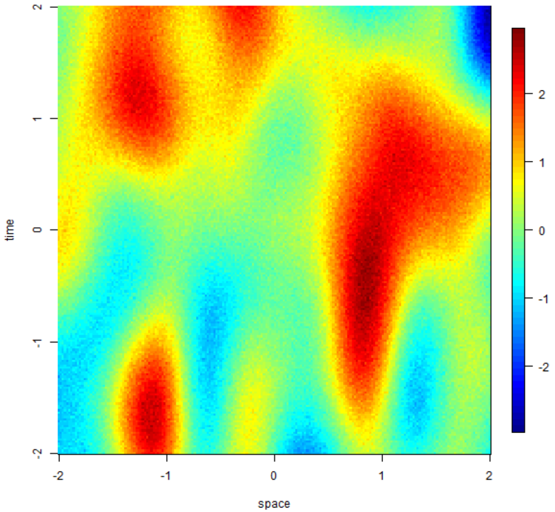
$$k_\theta(\tau) = \sum_{q=1}^{Q} w_q \exp(-2\pi^2\tau^2 v_q) \cos(2\pi\tau\mu_q)$$

- Answer 1: empirical Bayes aka maximize the marginal likelihood

$$\arg\max_\theta p(y|\theta) = \arg\max_\theta \int p(y|\boldsymbol{f})p(\boldsymbol{f}|\theta)d\boldsymbol{f}$$

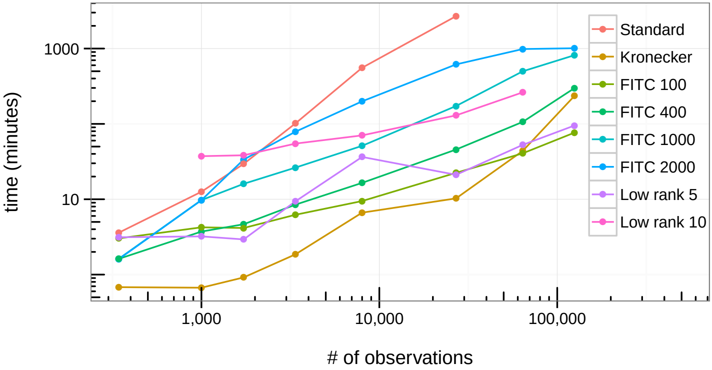
- Answer 2: fully Bayesian inference, place priors on hyperparameters, use MCMC

Experiments



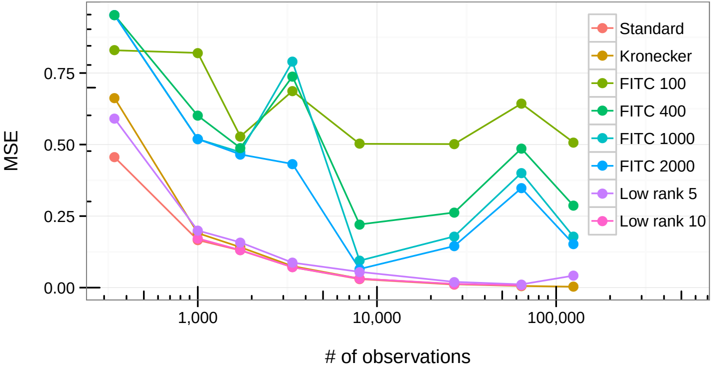
Experiments: Kronecker with Laplace

Run-time of our algorithm vs. competitors

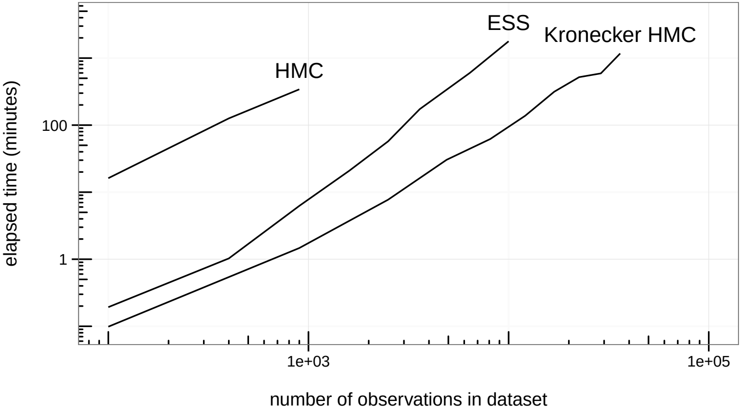


Experiments: Kronecker with Laplace

Accuracy of our algorithm vs. competitors



Experiments: Kronecker with MCMC



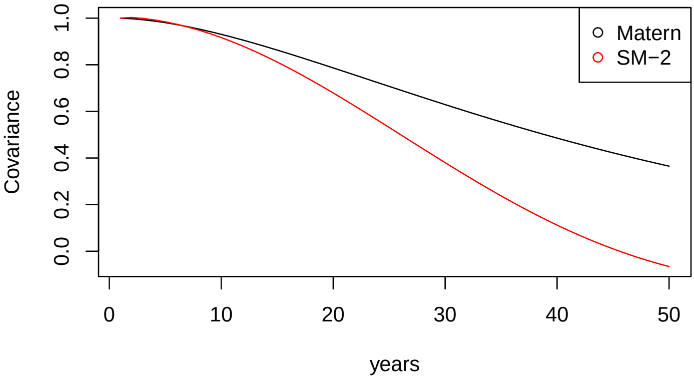
Real data: disease incidence

- Measles incidence $K_s \otimes K_t$ yearly for 50 states, 1935-1965 (n = 1550) from Project Tycho[1]
- Fit with Laplace approximation (learn hyperparameters by maximizing the marginal likelihood)
- K_s is Matérn-3/2, K_t is either Matérn-5/2 or SM-2

Method	Matérn	SM-2
Run-time	**4.4 minutes**	6 minutes
RMSE	8680	**1977**
Log-lik.	-14039	**-12869**

[1] tycho.pitt.edu

Results



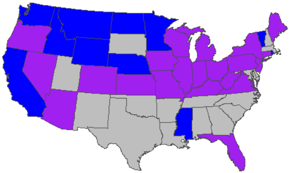
Real data: sampling

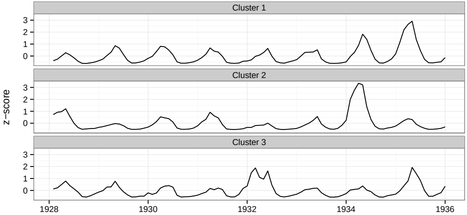
- Time series of monthly population-adjusted incidence of hepatitis A, measles, mumps, pertussis and rubella from Project Tycho
- Categorical data: $K_t \otimes K_c$ where K_c is a cross-covariance matrix over diseases with a uniform prior (actually, Lkj prior)

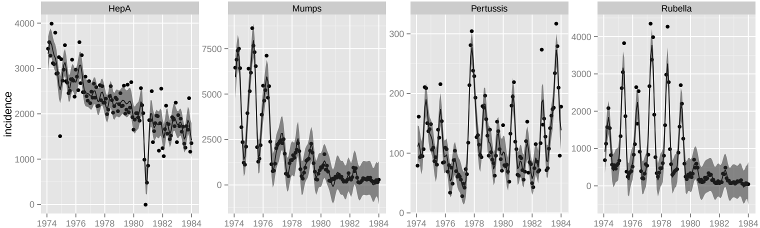
	Hepatitis A	Mumps	Pertussis	Rubella
Hepatitis A	1	0.6 (0.4,0.8)	-0.3 (-0.6,-0.1)	0.4 (0.1,0.6)
Mumps		1	-0.2 (-0.4,0.0)	0.6 (0.4,0.7)
Pertussis			1	-0.2 (-0.5,-0.0)
Rubella				1

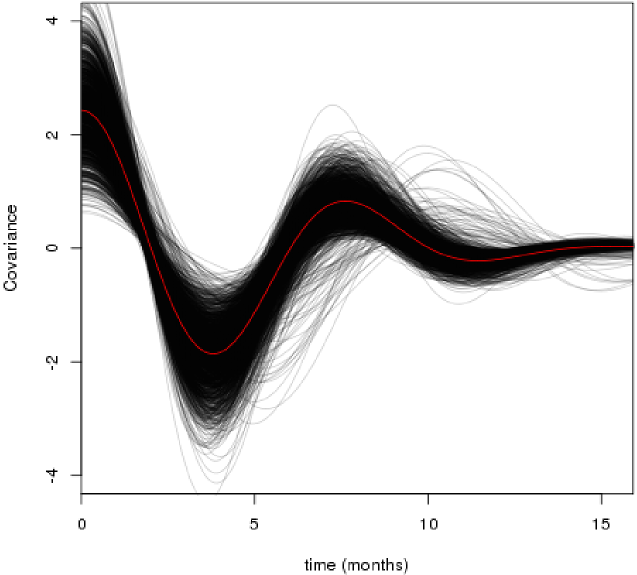
Real data: sampling

Factor analysis: $K_t \otimes K_s$ where $K_s = LL^\top + \sigma^2 I$, rows of L have a Dirichlet prior








Covariance
4
2
0
-2
-4
0
5
10
15
time (months)

Conclusion[3]

- Motivated use of GPs for spatiotemporal modeling
- Many settings match the Kronecker / grid structure
- Fully Bayesian approach: priors over kernel hyperparameters, missing data, complex models, implemented in Stan (source code in Appendix to paper on my website)
- Approximate Laplace approach is part of latest version of GPML[2] package
- Future work: more efficient MC inference for non-Gaussian likelihoods, variational inference (in Stan!)

[2] www.gaussianprocess.org/gpml/code

[3] Funding acknowledgement: NSF grant IIS-0953330

Non-Gaussian likelihoods: Inference

$$p(\boldsymbol{f}|\boldsymbol{y}, X) \approx \mathcal{N}(\boldsymbol{f}|\hat{\boldsymbol{f}}, (K^{-1} + W)^{-1})$$

$$\text{for } W = -\nabla\nabla \log p(\boldsymbol{y}|\boldsymbol{f}).$$

- The problem: covariance in Laplace approximation $(K^{-1} + W)^{-1}$ is not Kronecker
- Matrix inverse with LCG: matrix-vector multiplications are still fast
- Small number of evaluations required, each efficient:

$$\begin{aligned}
&(K^{-1} + W)v \\
&= K^{-1}v + Wv \\
&= (K_1^{-1} \otimes K_2^{-1})v + Wv
\end{aligned}$$

Non-Gaussian likelihoods: Learning

Laplace approximate marginal likelihood:

$$\log p(\boldsymbol{y}|X, \boldsymbol{\theta}) = \log \int \exp[\Psi(\boldsymbol{f})] d\boldsymbol{f}$$
$$\approx \log p(\boldsymbol{y}|\hat{\boldsymbol{f}}) - \frac{1}{2}\boldsymbol{\alpha}^\top K^{-1}\boldsymbol{\alpha} - \frac{1}{2}\log|I + KW|,$$

Tricky term: $\log|I + KW|$. For psd matrices U and V, Fiedler [1971]:

$$\prod_i (u_i + v_i) \leq |U + V| \leq \prod_i (u_i + v_{n-i+1})$$

where $u_1 \leq u_2 \leq \ldots \leq u_n$ and $v_1 \leq \ldots \leq v_n$ are the eigenvalues of U and V.

Fiedler bound

K has eigenvalues $e_1 \leq e_2 \leq \ldots \leq e_n$.
W has eigenvalues $w_1 \leq w_2 \leq \ldots \leq w_n$.

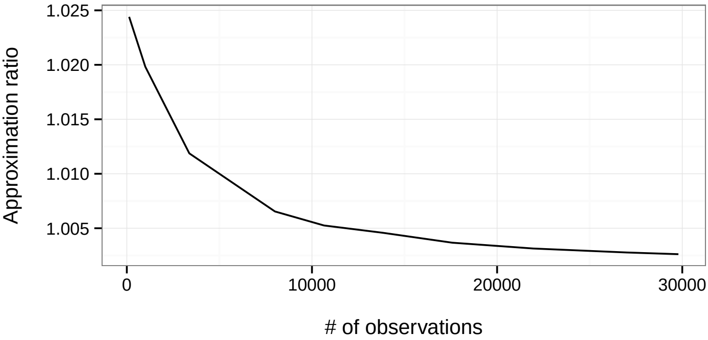
$$\begin{aligned} \log|I + KW| &= \log(|K + W^{-1}||W|) \\ &\leq \log \prod_i (e_i + w_i^{-1}) \prod_i w_i \\ &= \sum_i \log(1 + e_i w_i) \end{aligned}$$

Final bound on log-marginal likelihood:

$$\log p(\boldsymbol{y}|X, \boldsymbol{\theta}) \geq \log p(\boldsymbol{y}|\hat{\boldsymbol{f}}) - \frac{1}{2}\hat{\boldsymbol{\alpha}}^\top K^{-1}\hat{\boldsymbol{\alpha}} - \frac{1}{2}\sum_i \log(1 + e_i w_i)$$

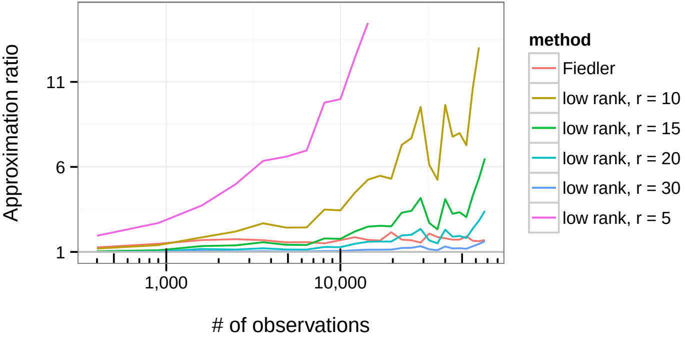
Experiments: synthetic data

Accuracy of our marginal likelihood approximation



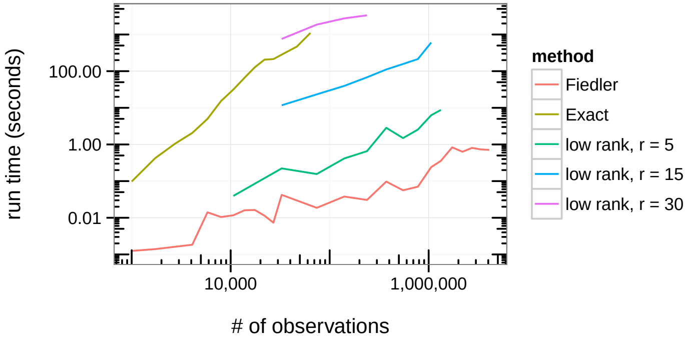
Experiments: synthetic data

Accuracy of our log-determinant approximation



Experiments: synthetic data

Run-time of our log-determinant approximation



Laplace approximation

- Posterior inference: $p(\boldsymbol{f}|\boldsymbol{y}, X) \propto p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|X)$
- Newton's method to find $\hat{\boldsymbol{f}}$
- Taylor expansion of log posterior at $\hat{\boldsymbol{f}}$
- The result is a Gaussian approximation

$$p(\boldsymbol{f}|\boldsymbol{y}, X) \approx \mathcal{N}(\boldsymbol{f}|\hat{\boldsymbol{f}}, (K^{-1} + W)^{-1})$$

for $W = -\nabla\nabla \log p(\boldsymbol{y}|\boldsymbol{f})$.

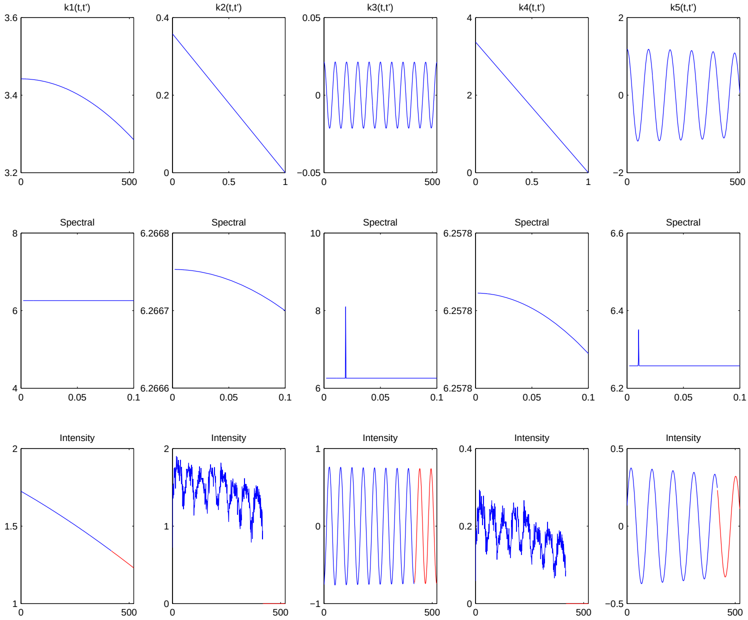
Kronecker methods for non-Gaussian likelihoods with Laplace approximation

$$p(\boldsymbol{f}|\boldsymbol{y}, X) \approx \mathcal{N}(\boldsymbol{f}|\hat{\boldsymbol{f}}, (K^{-1} + W)^{-1})$$

$$\text{for } W = -\nabla\nabla \log p(\boldsymbol{y}|\boldsymbol{f}).$$

- The problem: covariance in Laplace approximation $(K^{-1} + W)^{-1}$ is not Kronecker
- Matrix inverse with LCG: matrix-vector multiplications are still fast
- Upper-bound log-determinant using eigenvalues of K and W (diagonal)

Results


k1(t,t')
k2(t,t')
k3(t,t')
k4(t,t')
k5(t,t')
Spectral
Spectral
Spectral
Spectral
Spectral
Intensity
Intensity
Intensity
Intensity
Intensity

Source code

```
data {
  int<lower=1> n1;
  int<lower=1> n2;
  vector[n1] x1;
  vector[n2] x2;
  matrix[n1,n2] y;
  real sigma2;
}

parameters {
  real<lower=0> bw1;
  real<lower=0> bw2;
  real<lower=0> var1;
}
```

Source code

```
model {
  matrix[n1, n1] Sigma1;
  matrix[n2, n2] Sigma2;
  matrix[n1, n1] Q1;
  matrix[n2, n2] Q2;
  vector[n1] L1;
  vector[n2] L2;
  matrix[n1,n2] eigenvalues;

  for (i in 1:n1) {
    Sigma1[i, i] <- var1;
    for (j in (i+1):n1) {
      Sigma1[i, j] <- var1 * exp(-(x1[i]-x1[j])^2*bw1);
      Sigma1[j, i] <- Sigma1[i, j];
    }
  }
  for (i in 1:n2) {
    Sigma2[i, i] <- 1;
    for (j in (i+1):n2) {
      Sigma2[i, j] <- exp(-(x2[i]-x2[j])^2*bw2);
      Sigma2[j, i] <- Sigma2[i, j];
    }
  }

  Q1 <- eigenvectors_sym(Sigma1);
  Q2 <- eigenvectors_sym(Sigma2);
  L1 <- eigenvalues_sym(Sigma1);
  L2 <- eigenvalues_sym(Sigma2);

  eigenvalues <- calculate_eigenvalues(L1,L2,n1,n2,sigma2);
  var1 ~ lognormal(0,1);
  bw1 ~ cauchy(0,2.5);
  bw2 ~ cauchy(0,2.5);
  sigma2 ~ lognormal(0,1);
  increment_log_prob(-0.5 * sum(y .* kron_mvprod(Q1,Q2,
```