Scalable Gaussian Process Analysis – the SCALAGauss Project

Mihai Anitescu (MCS, Argonne),

With, Jie Chen, Emil Constantinescu (ANL); Michael Stein (Chicago)

Web site of project [*http://press3.mcs.anl.gov/scala-gauss/*](http://press3.mcs.anl.gov/scala-gauss/)

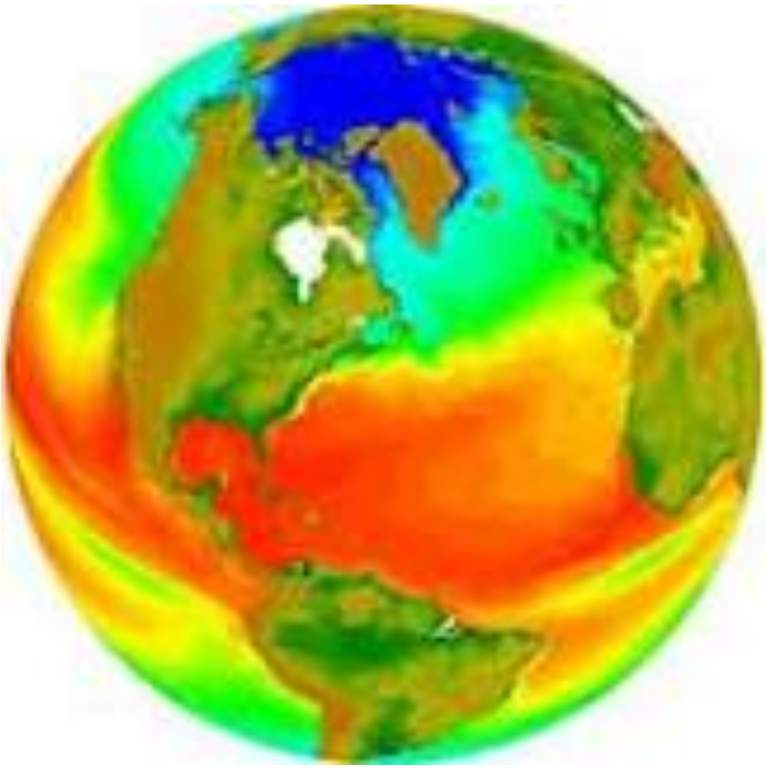
VERSION OF 5/1/2012 *At SAMSI-HPC-UQ workshop Oak Ridge, May 1, 2012*

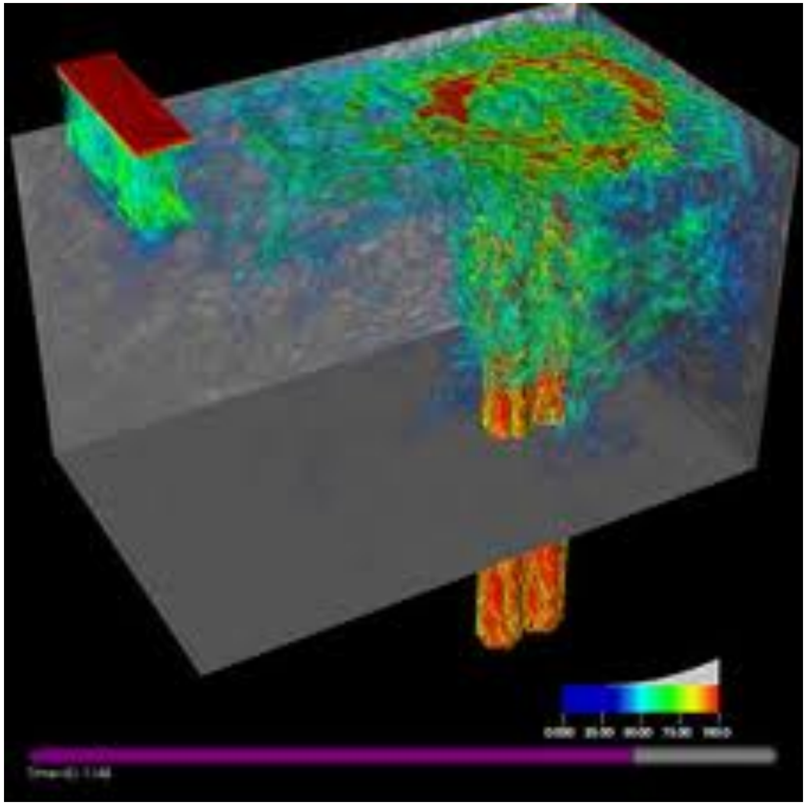
Outline

- 1. Context
- 2. Scalable Max Likelihood Calculations with GPs
- 3. Linear Algebra, Preconditioning
- 4 . Scalable Matrix-Vector Multiplication with Covariance Matrices
- 5. Scalable Sampling Methods for Gaussian Processes.
- 6. Physics-based cross-covariance models

1. CONTEXT

Application: Interpolation with UQ of Spatio-Temporal Processes





Source: http://www.ccs.ornl.gov/

Gaussian process regression (kriging): Setup

- Gaussian process (GP): $f(x) \sim \mathcal{N}(m(x), k(x,x'))$

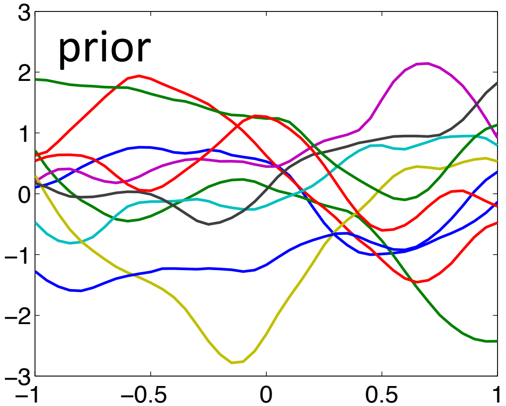
$$\mathrm{E}\{f(x)\} = \overline{f(x)} = m(x)$$
$$\mathrm{Cov}(f(x)) = k(x,x')$$

- **Most common: Stationary** $k(x,x') = k(x - x')$
- Data (observations)/predictions: $y = f(x) + \varepsilon \ / \ y_* = f(x_*)$
- GP joint distribution:

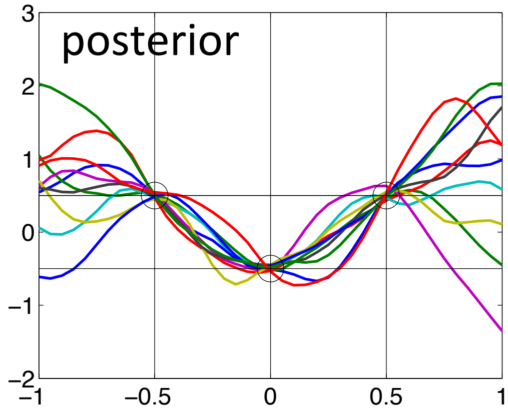
$$\begin{bmatrix} y \\ y_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mathbf{m}(X) \\ \mathbf{m}(X_*) \end{bmatrix}, \left[\begin{array}{c|c} \mathbf{K}_{11} + \Sigma & \mathbf{K}_{12} \\ \hline \mathbf{K}_{21} & \mathbf{K}_{22} \end{array} \right] \right)$$

- Predictive distribution:

$$\overline{\mathbf{y}_*|\mathbf{X},\mathbf{X}_*,\mathbf{y}} = \mathbf{m}(X_*) + \mathbf{K}_{21}\left(\mathbf{K}_{11} + \Sigma\right)^{-1}\left(\mathbf{y} - \mathbf{m}(X)\right)$$
$$\mathrm{Cov}(\mathbf{y}_*|\mathbf{X},\mathbf{X}_*,\mathbf{y}) = \mathbf{K}_{22} - \mathbf{K}_{21}\left(\mathbf{K}_{11} + \Sigma\right)^{-1}\mathbf{K}_{12}$$

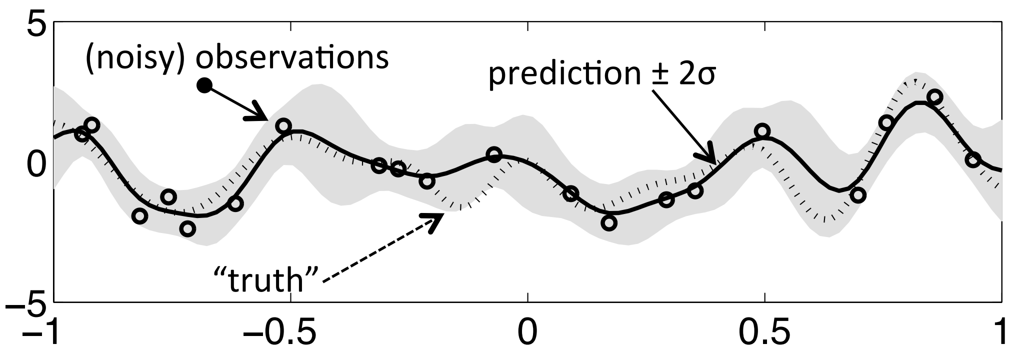


$$y = [\ 0.5 \quad -0.5 \quad 0.5\ ]$$
$$x = [\ -0.5 \quad 0 \quad 0.5\ ]$$



Gaussian process regression (kriging): inferences

- GP regression or kriging: related to autoregressive models, Kalman filtering



- Covariance function (kernel):

$$K(p,q) = k(d) = \sigma^2 \exp\left(-\frac{d^2}{2\ell^2}\right),$$

$$d = |x_p - x_q|$$

- Matérn covariance kernel:

$$k(d) = \sigma^2 \frac{2^{1-\frac{\nu}{2}}}{\Gamma(\nu)} \left(\frac{d\sqrt{\nu}}{\ell}\right)^{\nu} K_\nu\left(\frac{\sqrt{2}d\sqrt{\nu}}{\ell}\right)$$

- Marginal likelihood:

$$\mathcal{P}(y|x) = \int \mathcal{P}(y|f,x)\,\mathcal{P}(f|x)\,df$$

- Log- marginal likelihood:

$$\log(\mathcal{P}(y|X;\theta)) = -\frac{1}{2} y^T \left(K_{11}(\theta) + \Sigma\right)^{-1} y - \frac{1}{2}\log|K_{11}(\theta) + \Sigma|$$

MLE-II $\theta = [\sigma^2,\ \ell^2, \ldots]^T; \theta^* = \arg\max(\log(\mathcal{P}(y|X;\theta)))$

What makes a covariance function acceptable? Bochner's theorem (this slide from Rasmussen)

Theorem 4.1 *(Bochner's theorem) A complex-valued function k on $\mathbb{R}^D$ is the covariance function of a weakly stationary mean square continuous complex-valued random process on $\mathbb{R}^D$ if and only if it can be represented as*

$$k(\boldsymbol{\tau}) = \int_{\mathbb{R}^D} e^{2\pi i \mathbf{s}\cdot\boldsymbol{\tau}}\, d\mu(\mathbf{s}) \qquad (4.5)$$

where μ is a positive finite measure. $\square$

- This defines the **spectral density** of a covariance process (i.e. its FFT, which must be real and nonnegative everywhere).
- Some example processes and densities

Square Exponential

$$k_{\mathrm{SE}}(r) = \exp\left(-\frac{r^2}{2\ell^2}\right),$$

$$(2\pi\ell^2)^{D/2}\exp(-2\pi^2\ell^2 s^2).$$

Matern

$$k_{\mathrm{Matern}}(r) = \frac{2^{1-\nu}}{\Gamma(\nu)}\left(\frac{\sqrt{2\nu}r}{\ell}\right)^{\nu} K_\nu\left(\frac{\sqrt{2\nu}r}{\ell}\right),$$

$$S(s) = \frac{2^D\pi^{D/2}\Gamma(\nu+D/2)(2\nu)^\nu}{\Gamma(\nu)\ell^{2\nu}}\left(\frac{2\nu}{\ell^2}+4\pi^2 s^2\right)^{-(\nu+D/2)}$$

Matern 3/2

$$k_{\nu=3/2}(r) = \left(1+\frac{\sqrt{3}r}{\ell}\right)\exp\left(-\frac{\sqrt{3}r}{\ell}\right),$$

Tasks and challenges

- Sampling
- Maximum likelihood
- Interpolation/Kriging (solving linear system with K)
- Regression/Classification (solving linear systems with K)
- …

$$\log(p(J|S;\theta) = -\frac{1}{2}Y^T K^{-1} Y + \frac{1}{2}Y^T K^{-1} H (H^T K^{-1} H)^{-1} H^T K Y - \frac{1}{2}\log|K| - \frac{m}{2}\log(2\pi)$$

$$K = A^T A, \quad \xi \sim N(0,I), \quad y = M + A\xi \sim N(m,K)$$

- A lot of the basic tasks require matrix computations w.r.t. the covariance matrix K (and most often, Cholesky).
- But for 1B data points, you need 8*10^18 bytes to store = 8 EXABYTES, so cannot store K.
- How do you do compute log-det and A without storing the covariance matrix? And Hopefully in O(number data points) operations?
- The same challenges appear even outside GPs, as soon as you need to deal with full correlation.

2. SCALABLE MAXIMUM LIKELIHOOD CALCULATIONS

Maximum Likelihood Estimation (MLE)

- A family of covariance functions parameterized by θ: ϕ(x; θ)
- Maximize the log-likelihood to estimate θ:

$$
\begin{aligned}
\max_{\theta} L(\theta) &= \log\left\{(2\pi)^{-n/2}(\det K)^{-1/2}\exp(-y^T K^{-1} y/2)\right\} \\
&= -\frac{1}{2} y^T K^{-1} y - \frac{1}{2}\log(\det K) - \frac{n}{2}\log 2\pi
\end{aligned}
$$

- First order optimality: (also known as score equations)

$$
\frac{1}{2} y^T K^{-1}(\partial_j K) K^{-1} y - \frac{1}{2}\mathrm{tr}\left[K^{-1}(\partial_j K)\right] = 0
$$

Maximum Likelihood Estimation (MLE)

The log-det term poses a significant challenge for large-scale computations

$$\max_{\theta} \quad -\frac{1}{2} y^T K^{-1} y - \frac{1}{2} \log(\det K) - \frac{n}{2} \log 2\pi$$

- Cholesky of K: Prohibitively expensive!
- log(det K) = tr(log K): Need some matrix function methods to handle the log
- No existing method to evaluate the log-det term in sufficient accuracy

Sample Average Approximation of Maximum Likelihood Estimation (MLE)

We consider approximately solving the first order optimality instead:

$$\frac{1}{2} y^T K^{-1} (\partial_j K) K^{-1} y - \frac{1}{2} \mathrm{tr}\left[K^{-1} (\partial_j K) \right]$$

$$\approx \frac{1}{2} y^T K^{-1} (\partial_j K) K^{-1} y - \frac{1}{2N} \sum_{i=1}^{N} u_i^T \left[K^{-1} (\partial_j K) \right] u_i = 0$$

- A randomized trace estimator tr(A) = E[$u^T Au$]
  - u has i.i.d. entries taking ±1 with equal probability
- As N tends to infinity, the solution approaches the true estimate
- The variance introduced in approximating the trace is comparable with the variance of the sample y
  - So the approximation does not lose too much accuracy

- Numerically, one must solve linear systems with O(N) right-hand sides.

Stochastic Approximation of Trace

- When entries of u are i.i.d. with mean zero and covariance I

$$\mathrm{tr}(A) = E_u\left[u^T A u\right]$$

- The estimator has a variance

$$\mathrm{var}\left\{u^T A u\right\} = \sum_i \left(E\left[u_i^4\right] - 1\right) A_{ii}^2 + \frac{1}{2}\sum_{i \neq j}(A_{ij} + A_{ji})^2$$

- If each entry of u takes ±1 with equal probability, the variance is the smallest

$$\mathrm{var}\left\{u^T A u\right\} = \frac{1}{2}\sum_{i \neq j}(A_{ij} + A_{ji})^2$$

Convergence of Stochastic Programming - SAA

- Let

$$\theta: \text{ truth}$$

$$\hat{\theta}: \text{ sol of } \frac{1}{2} y^T K^{-1}(\partial_j K) K^{-1} y - \frac{1}{2}\mathrm{tr}\left[K^{-1}(\partial_j K)\right] = 0$$

$$\hat{\theta}^N: \text{ sol of } F = \frac{1}{2} y^T K^{-1}(\partial_j K) K^{-1} y - \frac{1}{2N}\sum_{i=1}^{N} u_i^T \left[K^{-1}(\partial_j K)\right] u_i = 0$$

- First result:

$$[V^N]^{-1/2}(\hat{\theta}^N - \hat{\theta}) \xrightarrow{D} \text{standard normal}, \quad V^N = [J^N]^{-T}\Sigma^N [J^N]^{-1}$$

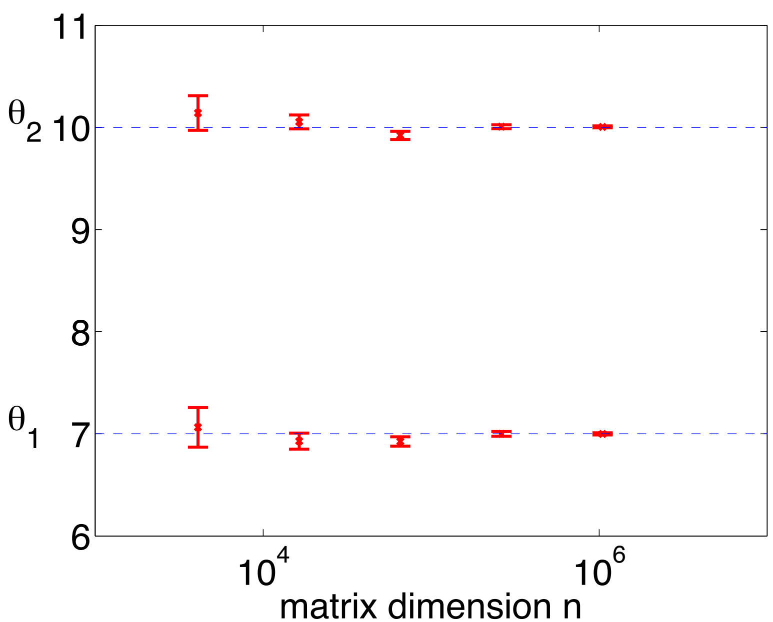
where

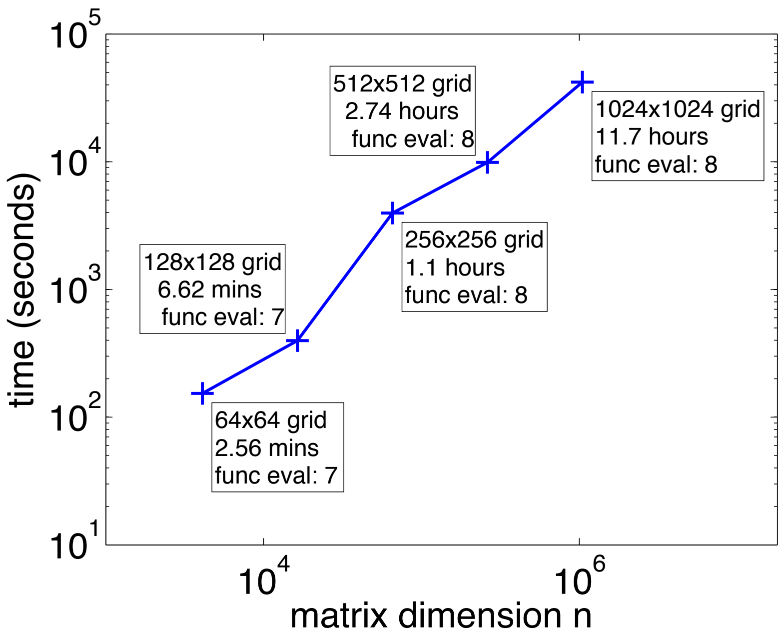
$$J^N = \nabla F(\hat{\theta}^N) \quad \text{and} \quad \Sigma^N = \mathrm{cov}\{F(\hat{\theta}^N)\}$$

- Note: Σ^N decreases in $O(N^{-1})$

Simulation: We scale

- Truth θ = [7, 10], Matern ν = 1.5





"Optimal" Convergence

- Let

$$\theta : \text{ truth}$$

$$\hat{\theta} : \text{ sol of } \frac{1}{2} y^T K^{-1} (\partial_j K) K^{-1} y - \frac{1}{2} \mathrm{tr}\left[K^{-1}(\partial_j K)\right] = 0$$

$$\hat{\theta}^N : \text{ sol of } F = \frac{1}{2} y^T K^{-1} (\partial_j K) K^{-1} y - \frac{1}{2N} \sum_{i=1}^{N} u_i^T \left[K^{-1}(\partial_j K)\right] u_i = 0$$

- Second result:

$$C^{-1/2}(\hat{\theta}^N - \theta) \xrightarrow{D} \text{standard normal}, \quad \mathrm{C} = A^{-T} B A^{-1}$$

where

$$-A = I, \text{ Fisher matrix} \quad \text{and} \quad B = I + \frac{1}{4N} J$$

- Note: J has a bound $J \le I \cdot \frac{[\mathrm{cond}(K)+1]^2}{\mathrm{cond}(K)}$, so C converges to I^{-1} in $O(N^{-1})$ if condition number of K is bounded.

3. LINEAR ALGEBRA; PRECONDITIONING

LINEAR ALGEBRA CHALLENGES: PRECONDITIONING AND MATRIX VECTOR MULTIPLICATIONS

We reduced max likelihood calculations to solving linear systems with K.

We next focus on the linear algebra:

- Preconditioning K
- Matrix-vector multiplication with K
- Solving linear system w.r.t. K with multiple right-hand sides

Covariance Model

- Matern covariance function

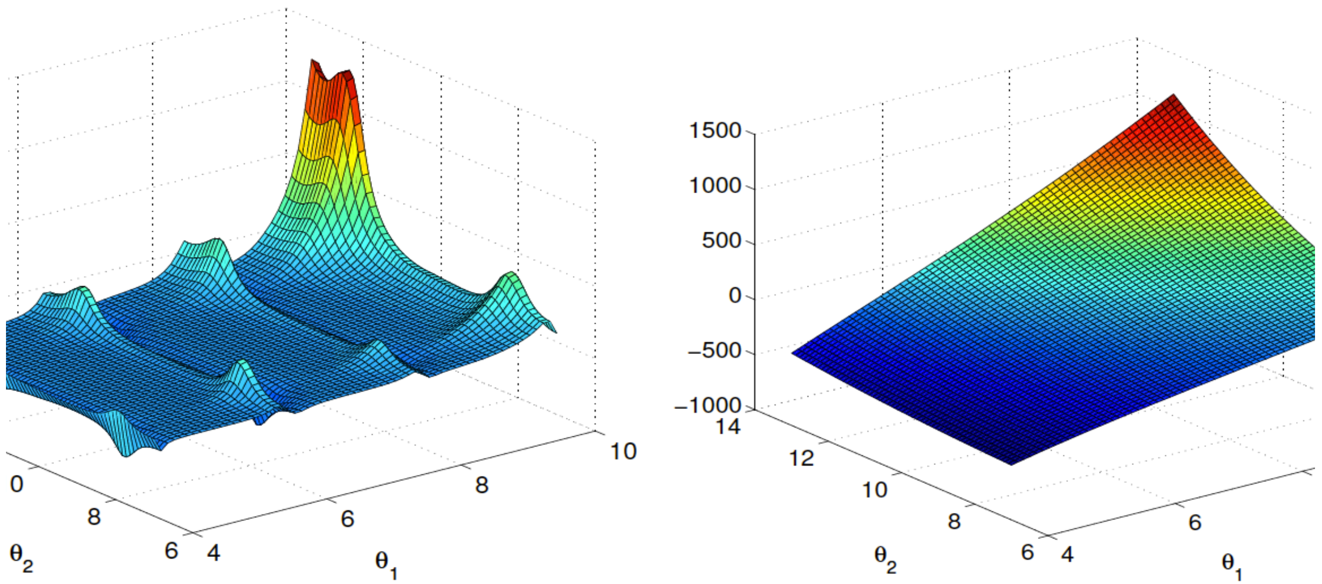
$$\phi(x) = \frac{1}{2^{\nu-1}\Gamma(\nu)}\left(\sqrt{2\nu r}\right)^{\nu} \mathrm{K}_{\nu}\left(\sqrt{2\nu r}\right) \quad \text{where} \quad r = \sqrt{\sum_{j=1}^{d} \frac{x_j^2}{\theta_j^2}}$$

  - ν: Example values 0.5, 1, 1.5, 2
  - θ: Scale parameters to estimate
  - K_ν is the modified Bessel function of the second kind of order ν
- Commonly used in spatial/temporal data modeling.
- The parameter ν is used to model the data with a certain level of smoothness.
- When ν -> ∞, the kernel is the Gaussian kernel.
- Spectral density

$$f(\omega) \propto \left(2\nu + \rho^2\right)^{-(\nu + d/2)} \quad \text{where} \quad \rho = \sqrt{\sum_{j=1}^{d} (\theta_j \omega_j)^2}$$

Why the Matern Kernel?

- In machine learning, people tend to use the square exponential kernel a lot.
- This assumes that all realizations are infinitely smooth, a fact rarely supported by data, especially high resolution data.
- The Matern Kernel allows one to adjust smoothness.
- The resulting covariance matrix is dense, compared to compact Kernels, but the likelihood surface is much smoother.



(a) Compact kernel.

(b) Matern kernel.

Condition Number

- K is increasingly ill-conditioned.
- If the grid is in a fixed, finite domain $\subset \mathrm{R}^d$, then cond(K) = O($n^{2v/d+1}$)

- On regular grid, K is (multi-level) Toeplitz, hence a circulant preconditioner applies

$$\begin{bmatrix} t_0 & t_{-1} & \cdots & t_{-n+2} & t_{-n+1} \\ t_1 & t_0 & t_{-1} & & t_{-n+2} \\ \vdots & t_1 & t_0 & \ddots & \vdots \\ t_{n-2} & & \ddots & \ddots & t_{-1} \\ t_{n-1} & t_{n-2} & \cdots & t_1 & t_0 \end{bmatrix} \longrightarrow \begin{bmatrix} c_0 & c_{n-1} & \cdots & c_2 & c_1 \\ c_1 & c_0 & c_{n-1} & & c_2 \\ \vdots & c_1 & c_0 & \ddots & \vdots \\ c_{n-2} & & \ddots & \ddots & c_{n-1} \\ c_{n-1} & c_{n-2} & \cdots & c_1 & c_0 \end{bmatrix}$$

- More can be done by considering filtering

Condition Number

- Filtering (1D): if $f(w)w^2$ bounded away from 0 and ∞ as w -> ∞
- Let $0 \le x_0 \le x_1 \le \ldots \le x_n \le T$. $d_j = x_j - x_{j-1}$.

$$Y_j^{(1)} = [Z(x_j) - Z(x_{j-1})] / \sqrt{d_j}, \qquad K^{(1)}(j,l) = \operatorname{cov}\left\{Y_j^{(1)}, Y_l^{(1)}\right\}$$

- Then $K^{(1)}$ has a bounded condition number independent of n

- Filtering (1D): if $f(w)w^4$ bounded away from 0 and ∞ as w -> ∞

$$Y_j^{(2)} = \frac{Z(x_{j+1}) - Z(x_j)}{2d_{j+1}\sqrt{d_{j+1} + d_j}} - \frac{Z(x_j) - Z(x_{j-1})}{2d_j\sqrt{d_{j+1} + d_j}}, \qquad K^{(2)}(j,l) = \operatorname{cov}\left\{Y_j^{(2)}, Y_l^{(2)}\right\}$$

- Then $K^{(2)}$ has a bounded condition number independent of n

Condition Number

- Filtering (high dimension, regular grid): if f(w) is asymptotically $(1+|w|)^{-4\tau}$

$$\Delta Z(x_j) = \sum_{p=1}^{d} Z(x_j - \delta e_p) - 2Z(x_j) + Z(x_j + \delta e_p)$$

$$K^{[\tau]}(j,l) = \mathrm{cov}\left\{\Delta^{[\tau]} Z(x_j), \Delta^{[\tau]} Z(x_l)\right\}$$

- Then $K^{[\tau]}$ has a bounded condition number independent of n

- Use the filter as a preconditioner

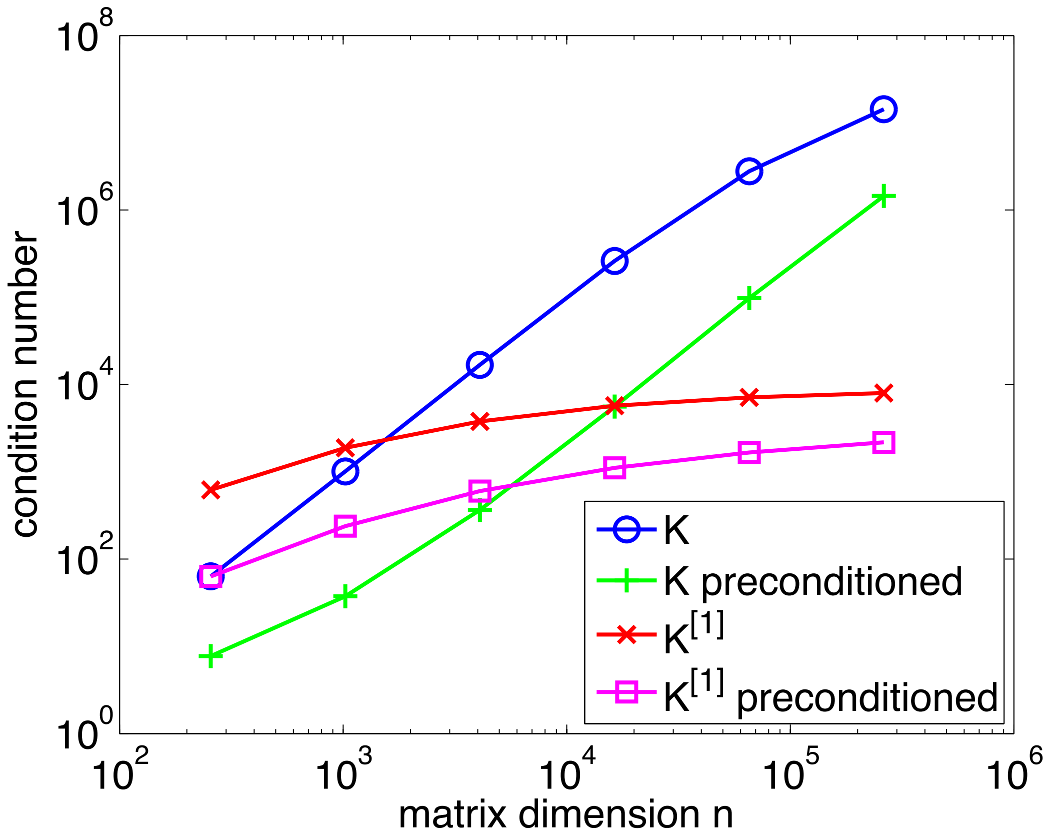
$$K^{[\tau]} = \left[L^{[\tau]}\right]\cdots\left[L^{[2]}\right]\left[L^{[1]}\right]K\left[L^{[1]}\right]^T\left[L^{[2]}\right]^T\cdots\left[L^{[\tau]}\right]^T$$

In 2D, L is the 5-point stencil matrix with rows w.r.t. the grid boundary removed.

- Similarly for the filters in the preceding slide

Condition Number

- Effect of filtering ($K^{[\tau]}$ can be further preconditioned by circulant preconditioner)



Block CG

- Preconditioned Conjugate Gradient (M is preconditioner)

$$Ax = b$$

$$x_{j+1} = x_j + \alpha_j p_j$$
$$r_{j+1} = r_j - \alpha_j A p_j$$
$$p_{j+1} = M r_{j+1} + \beta_j p_j$$

where

$$\alpha_j = r_j^T M r_j / p_j^T A p_j$$
$$\beta_j = r_{j+1}^T M r_{j+1} / r_j^T M r_j$$

$$AX = B \quad \text{(block version)}$$

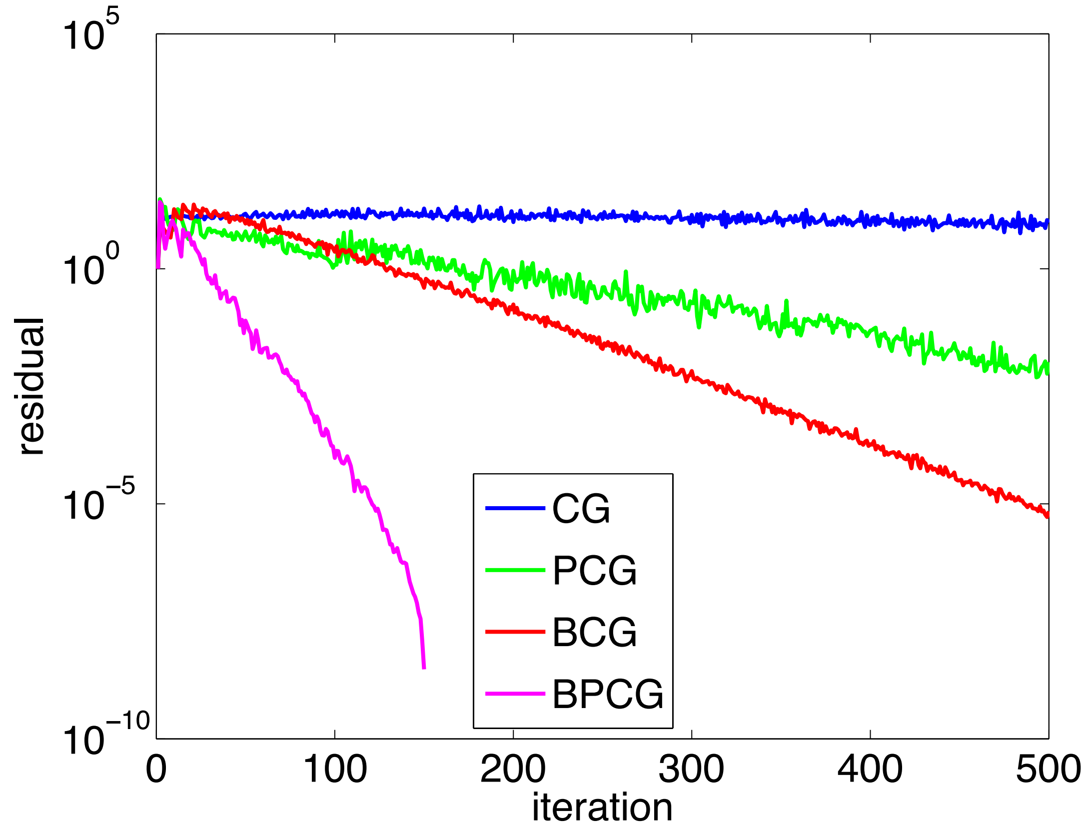
$$X_{j+1} = X_j + P_j \alpha_j$$
$$R_{j+1} = R_j - A P_j \alpha_j$$
$$P_{j+1} = (M R_{j+1} + P_j \beta_j) \gamma_{j+1}$$

where

$$\alpha_j = (P_j^T A P_j)^{-1} \gamma_j^T (R_j^T M R_j)$$
$$\beta_j = \gamma_j^{-1} (R_j^T M R_j)^{-1} (R_{j+1}^T M R_{j+1})$$

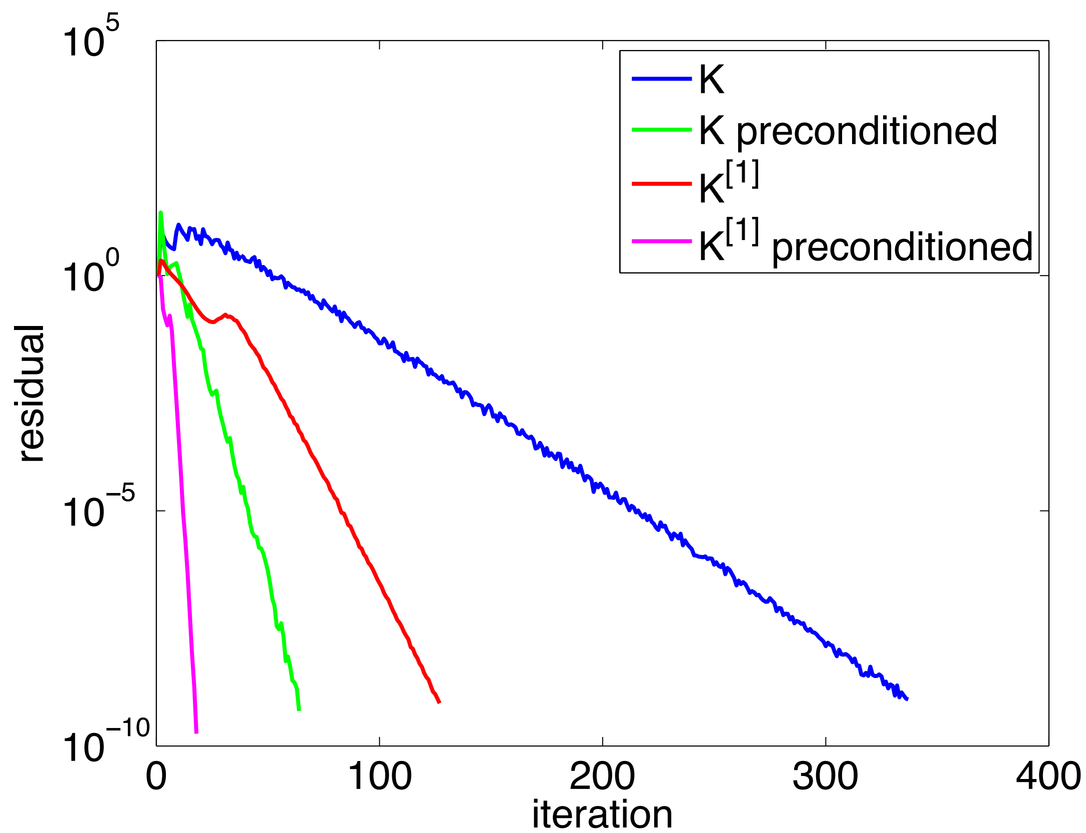
Block CG

- CG, block CG, and the preconditioned versions using circulant preconditioner



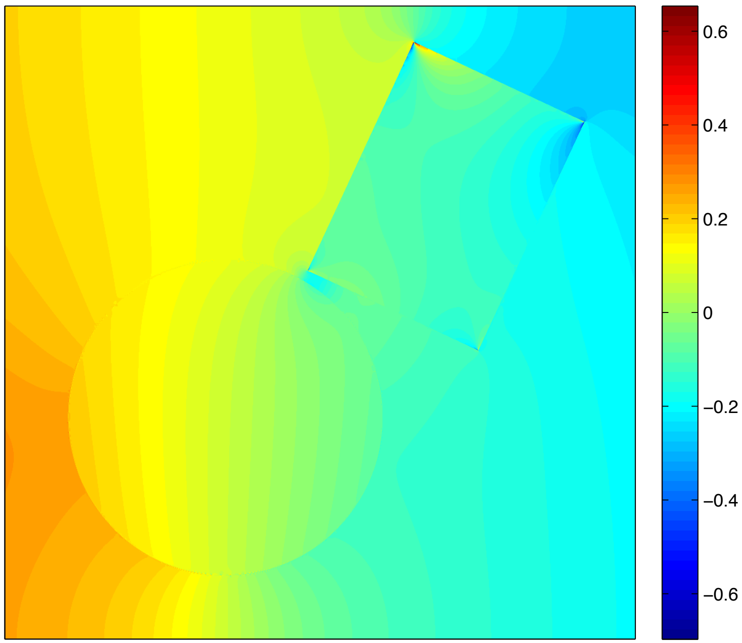
Experimental Results

- Combined effect of circulant preconditioning and filtering

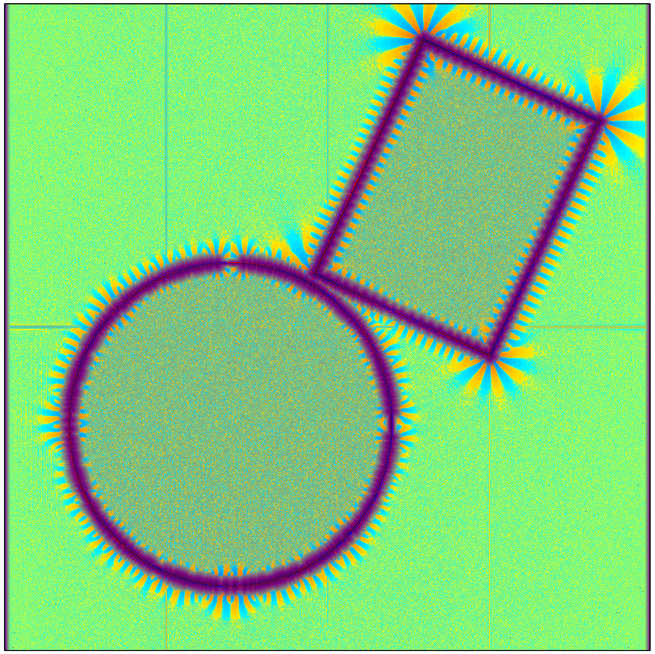


3.1 PHYSICS-INSPIRED “PROBLEM”

Stokes Flow



(a) Pressure field



(b) Filtered pressure field in log-scale

Stokes Flow

Fitted a power-law model $\phi(x;\alpha,C) = \Gamma(-\alpha/2)\cdot C\|x\|^{\alpha}$

Data	Circle	Rectangle	Boundary	Background
# Points	1.5e+5	7.9e+4	3.1e+4	4.7e+5
Fitted α	0.1819	0.3051	0.7768	1.5945
Fitted C	722.54	803.07	3309.4	626180.0
eig(Fisher^{-1})$^{1/2}$	5.04e-4	9.80e-4	2.50e-3	8.11e-4
	1.31e+1	1.86e+1	1.26e+2	5.44e+3

$$\frac{\sqrt{\lambda(V^{-1})}}{\sqrt{\lambda(\mathcal{I}^{-1})}} \qquad \begin{array}{cc} 1.0283 & 1.0289 \\ 1.0284 & 1.0284 \\ 1.0010 & 1.0009 \end{array}$$

Conclusion GP

- State-of-the-art methods use Cholesky to do sampling and solve ML.
  - Can probably handle data size up to $n = O(10^4)$ or $O(10^5)$.
- We propose a framework to overcome the Cholesky barrier.
  - Use a matrix-free method to do sampling.
  - Reformulate maximum likelihood using stochastic approximation.
  - Use iterative solver to solve linear systems.
  - Use a filtering technique to reduce the condition number.
- On going work
  - Investigating the scaling of parallel FFT for $n = O(10^6)$ and larger computations.
  - For scattered points, investigating a discrete Laplace operator for filtering.
  - Implementing a fast summation method to do mat-vec.
- Details: ScalaGauss project web site.