# Scalable Gaussian Process Computations Using Hierarchical Matrices

**C.J. Geoga**[1,2] M. Anitescu[1,3] M.L. Stein[3]

1: Mathematics and Computer Science Divison, Argonne National Laboratory
2: Department of Statistics, Rutgers University
3: Department of Statistics, University of Chicago

# Gaussian processes and maximum likelihood estimation

# Spatial processes

Many natural processes evolve over space in some way that depends on other locations.

Notation:

- Let $n$ denote the number of observations.
- At spatial location $\boldsymbol{x}$, denote the value of the process at $\boldsymbol{x}$ with $Z(\boldsymbol{x})$.
- Enumerate observations as $\boldsymbol{y} := \{y_j\}_{j=1}^n$ and $\{\boldsymbol{x}_j\}_{j=1}^n$
- Let $K$ denote the parametric kernel function used to model covariance with parameters $\boldsymbol{\theta}$:

$$\text{Cov}(y_j, y_k) = K(\boldsymbol{x}_j, \boldsymbol{x}_k; \boldsymbol{\theta}).$$

We require that $K$ be positive definite.

# Why model spatial processes?

- **Inference:** Processes often modeled as

$$Z(\boldsymbol{x}) := f_\theta(\boldsymbol{x}) + \varepsilon.$$

- **Interpolation:** Optimally "fill-in" missing values.

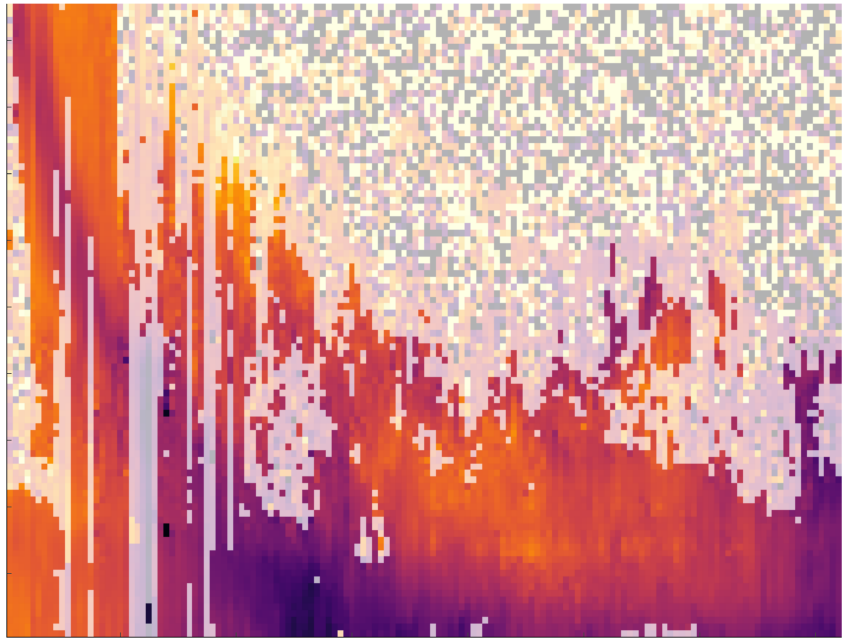


Figure: A field with missing values.

# Gaussian processes

If in the *process* model above $f_\theta(\boldsymbol{x}) = \mu(\boldsymbol{x})$ and $\varepsilon \sim \mathcal{N}(0, \boldsymbol{\Sigma})$, then the *data* is distributed as

$$\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}).$$

We call this a *Gaussian Process (GP) model*.

We often model the elements of $\boldsymbol{\Sigma}$ using a kernel function $K$, and are often parameterized in basic settings as

$$\boldsymbol{\Sigma}_{j,k} = K(\boldsymbol{x}_j, \boldsymbol{x}_k, \boldsymbol{\theta}) = \boldsymbol{\theta}_0 g\left(\frac{||\boldsymbol{x} - \boldsymbol{y}||}{\boldsymbol{\theta}_1}\right),$$

with $g$ a nice kernel function: $x \mapsto p_n(x)e^{-x}$, $x \mapsto (1+x)^{2k}$, etc.

# Why Gaussian processes models?

- **Easy to specify:** Just pick functions $\mu$ and $K$. $\mu$ often assumed to be zero!
- **Tractable math:** Easy expressions for interpolants, forecasts, etc.
- **Flexible:** Captures a huge variety of local and global behaviors.

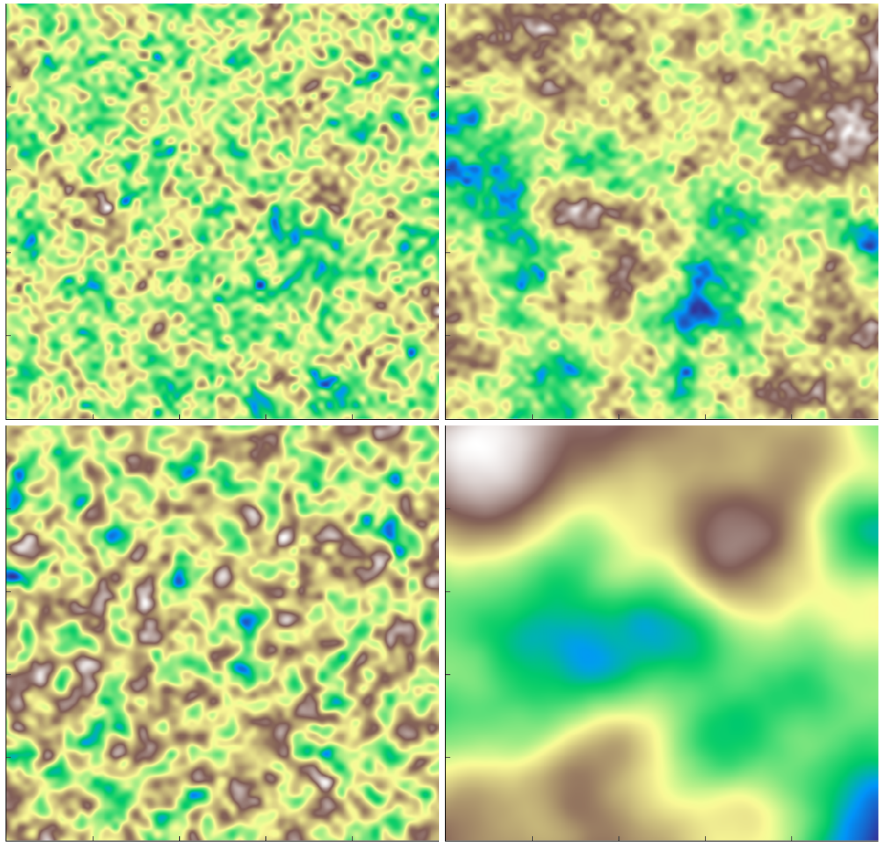


Figure: Some simulated processes.

# Estimation for Gaussian processes

Most often, we compute *Maximum Likelihood Estimators* (MLEs) by maximizing the log-likelihood. Here

$$l(\boldsymbol{\theta}) := -\frac{1}{2}\log|\boldsymbol{\Sigma}(\boldsymbol{\theta})| - \frac{1}{2}(\boldsymbol{y}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}(\boldsymbol{\theta})^{-1}(\boldsymbol{y}-\boldsymbol{\mu}) - \frac{n}{2}\log(2\pi).$$

Asymptotic theory suggests that MLEs have good statistical behavior for large sample sizes (Stein 1999).

For the rest of this talk, assume $\boldsymbol{\mu} \equiv \mathbf{0}$, let $\boldsymbol{\Sigma}$ always denote $\boldsymbol{\Sigma}(\boldsymbol{\theta})$, and suppress the constant term at the end of $l(\boldsymbol{\theta})$.

**So: can we always compute the MLE?**

# Estimation for Gaussian processes

**...Not so easily**: The linear algebra required for evaluating $l(\theta)$ scales very poorly.

- cubic time complexity
- quadratic storage complexity
- difficult numerics with condition number of $\Sigma$
- difficulty with optimization for certain covariance functions

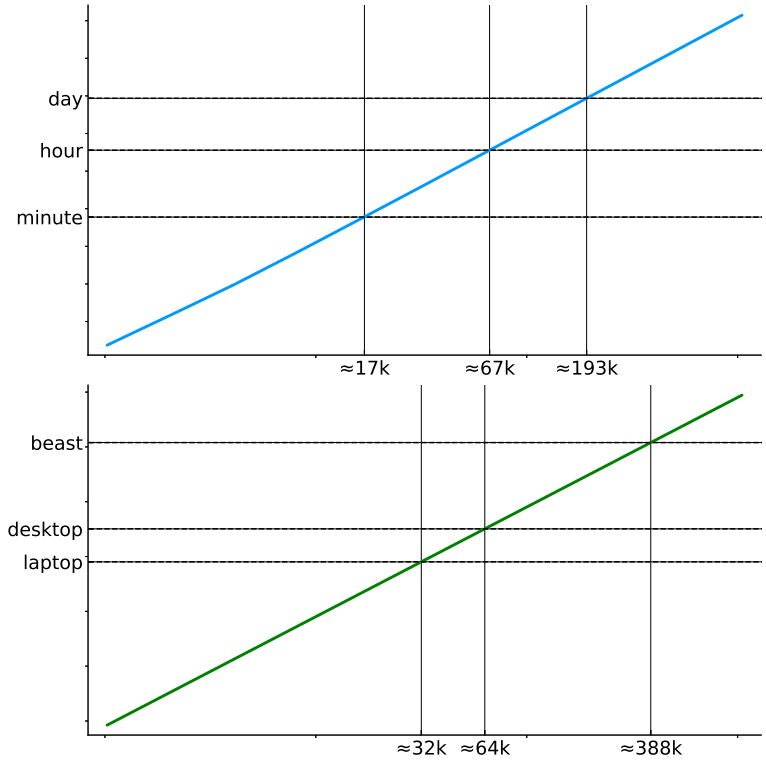



Figure: Sample time (top) and storage (bottom) complexity for LU factorization with some level perspectives.

# Some culture: historical attempts to compute or approximate MLEs

There have been many attempts to approximate $l(\boldsymbol{\theta})$ in a scalable way. Some highlights include:

- **Vecchia-type methods:** block sparsity in $\boldsymbol{\Sigma}$ (Vecchia 1988).
- **Markov Random Fields:** sparsity in $\boldsymbol{\Sigma}^{-1}$ (Rue and Held 2005).
- **Estimating equations:** solve the score equations

$$(\nabla l(\boldsymbol{\theta}))_j = -\frac{1}{2}\text{tr}\left(\boldsymbol{\Sigma}^{-1}\boldsymbol{\Sigma}_j\right) + \frac{1}{2}\boldsymbol{y}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\Sigma}_j\boldsymbol{\Sigma}^{-1}\boldsymbol{y},$$

where $\boldsymbol{\Sigma}_j := \frac{\partial}{\partial\theta_j}\boldsymbol{\Sigma}(\boldsymbol{\theta})$. Can use matrix free methods like circulant embedding or FMM (Anitescu, Chen, and Wang 2011; Stein, Chen, and Anitescu 2013).

# Hierarchical matrices, low-rank approximations, and likelihood approximations

# Low-rank structure of off-diagonal blocks

**A key insight:** for functions that are smooth away from the origin, off-diagonal blocks of resulting kernel matrices will have low numerical rank.

Some **very** brief history:

- Fast Multiple Method (Greengard and Rokhlin 1987)
  - matvec only
- Hierarchical matrices (Hackbusch 1999)
  - **direct methods for solve, determinant, factorization**
- Many others...

# A visualization:

$$\Sigma x \approx \begin{bmatrix} \begin{bmatrix} L_1 & U_{11}V_{11}^T \\ V_{11}U_{11}^T & L_2 \end{bmatrix} & U_{01}V_{01}^T \\ V_{01}U_{01}^T & \begin{bmatrix} L_3 & U_{12}V_{12}^T \\ V_{12}U_{12}^T & L_4 \end{bmatrix} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

- Easy to reason about complexity (*if rank structure is known*).
- This particular format is referred to as HODLR (Ambikasaran and Darve 2013), or *weakly admissible*. There are many other compression structures.

# How to compress low-rank off-diagonal blocks?

Let $\boldsymbol{A}_{I,J}$ denote the block of $\boldsymbol{A}$ corresponding to indices $I$ and $J$, and let $|I|$ denote the size of the index set.

- **Truncated SVD**:
  $A_{I,J} \approx U\Sigma V^T$, $U \in \mathbb{R}^{n \times p}$
  - optimal
  - slow
- **Adaptive Cross Approximation** (ACA) (Bebendorf 2000):
  $A_{I,J} \approx UV^T$, $U \in \mathbb{R}^{n \times p}$
  - fast, accurate, and adaptive
  - not differentiable
- **Nyström approximation** (Williams and Seeger 2001):
  $A_{I,J} \approx A_{I,P}A_{P,P}^{-1}A_{P,J}$, $|P| = p$
  - global, non-adaptive
  - guarantees positive-definiteness of approximation
  - differentiable

# A HODLR approximation for covariance matrices

Let $\widetilde{\Sigma}$ the HODLR approximation of $\Sigma$ with the Nyström approx. Our approximated likelihood looks like

$$\ell_H(\theta) := -\frac{1}{2}\log\left|\widetilde{\Sigma}\right| - \frac{1}{2}y^T\widetilde{\Sigma}^{-1}y.$$

$\widetilde{\Sigma}$ is differentiable, with $\widetilde{\Sigma}_{j,(I,J)} := \frac{\partial}{\partial\theta_j}\widetilde{\Sigma}_{I,J}(\theta)$ given by

$$\Sigma_{j,(I,P)}\Sigma^{-1}_{P,P}\Sigma_{P,J} - \Sigma_{I,P}\Sigma^{-1}_{P,P}\Sigma_{j,(P,P)}\Sigma^{-1}_{P,P}\Sigma_{P,J} + \Sigma_{I,P}\Sigma^{-1}_{P,P}\Sigma_{j,(P,J)}.$$

**$\widetilde{\Sigma}_j$ is a sum of 3 identically shaped HODLR matrices!**

Recall the gradient:

$$(\nabla l_H(\theta))_j = -\frac{1}{2}\mathrm{tr}\left(\widetilde{\Sigma}^{-1}\widetilde{\Sigma}_j\right) + \frac{1}{2}y^T\widetilde{\Sigma}^{-1}\widetilde{\Sigma}_j\widetilde{\Sigma}^{-1}y,$$

All we need is the trace of $\widetilde{\Sigma}^{-1}\widetilde{\Sigma}_j$...

# Stochastic trace estimation

Hutchinson (1990) gave us a stochastic trace estimator:

$$\operatorname{tr}\left(\widetilde{\boldsymbol{\Sigma}}^{-1}\widetilde{\boldsymbol{\Sigma}}_j\right) \approx \frac{1}{N}\sum_{j=1}^{N} \boldsymbol{u}^T\widetilde{\boldsymbol{\Sigma}}^{-1}\widetilde{\boldsymbol{\Sigma}}_j\boldsymbol{u}.$$

Since we can factorize $\widetilde{\boldsymbol{\Sigma}} = \boldsymbol{W}\boldsymbol{W}^T$, we can do better: the covariance matrix of the estimates

$$\operatorname{tr}\left(\widetilde{\boldsymbol{\Sigma}}^{-1}\widetilde{\boldsymbol{\Sigma}}_j\right) = \operatorname{tr}\left(\boldsymbol{W}^{-1}\widetilde{\boldsymbol{\Sigma}}_j\boldsymbol{W}^{-T}\right) \approx \frac{1}{N}\sum_{j=1}^{N} \boldsymbol{u}^T\boldsymbol{W}^{-1}\widetilde{\boldsymbol{\Sigma}}_j\boldsymbol{W}^{-T}\boldsymbol{u}$$

has a bound that is free of the condition number of $\widetilde{\boldsymbol{\Sigma}}$ (Stein, Chen, and Anitescu 2013)!

# A stable gradient estimator

Combining the exact derivative with the symmetrized trace estimator, we obtain a good estimator for $\nabla \ell_H(\boldsymbol{\theta})_j$:

$$\widehat{\nabla \ell_H(\boldsymbol{\theta})_j} = \underbrace{-\frac{1}{2N_h} \sum_{l=1}^{N_h} \boldsymbol{u}_l^T \boldsymbol{W}^{-1} \widetilde{\boldsymbol{\Sigma}}_j \boldsymbol{W}^{-T} \boldsymbol{u}_l}_{\text{quite a good approximation}} + \underbrace{\frac{1}{2} \boldsymbol{y}^T \widetilde{\boldsymbol{\Sigma}}^{-1} \widetilde{\boldsymbol{\Sigma}}_j \widetilde{\boldsymbol{\Sigma}}^{-1} \boldsymbol{y}}_{\text{exact}}.$$

Away from the MLE when the error in the trace term doesn't dominate the magnitude, this estimator reliably has relative error below 0.03%.

# A stable Hessian estimator

We also obtain a good estimator for the Hessian $H\ell_H(\boldsymbol{\theta})_j$:

$$\underbrace{\mathrm{tr}\left(\widetilde{\boldsymbol{\Sigma}}^{-1}\widetilde{\boldsymbol{\Sigma}}_j\widetilde{\boldsymbol{\Sigma}}^{-1}\widetilde{\boldsymbol{\Sigma}}_k\right)}_{\text{expected Fisher matrix } \mathcal{I}_{j,k}} - \underbrace{\mathrm{tr}\left(\widetilde{\boldsymbol{\Sigma}}^{-1}\widetilde{\boldsymbol{\Sigma}}_{jk}\right) + \frac{1}{2}\boldsymbol{y}^T\left(\frac{\partial}{\partial\theta_k}\widetilde{\boldsymbol{\Sigma}}^{-1}\widetilde{\boldsymbol{\Sigma}}_j\widetilde{\boldsymbol{\Sigma}}^{-1}\right)\boldsymbol{y}}_{\text{data-correction to expected information}}.$$

We can use the exact same methods to obtain a similar symmetrized estimator for the expected information and observed information (Hessian).

# Qasilinear complexity

**Under appropriate conditions, all of these expressions are computable in quasilinear complexity.**

- $p$ (rank of off-diagonal blocks) fixed
- Dyadic level grows with $\mathcal{O}(\log n)$.

But: we can't control pointwise precision $||\boldsymbol{\Sigma} - \widetilde{\boldsymbol{\Sigma}}||$.

So **is $\ell_H$ a useful approximation under the above conditions?**

# Numerical results

# Time complexity

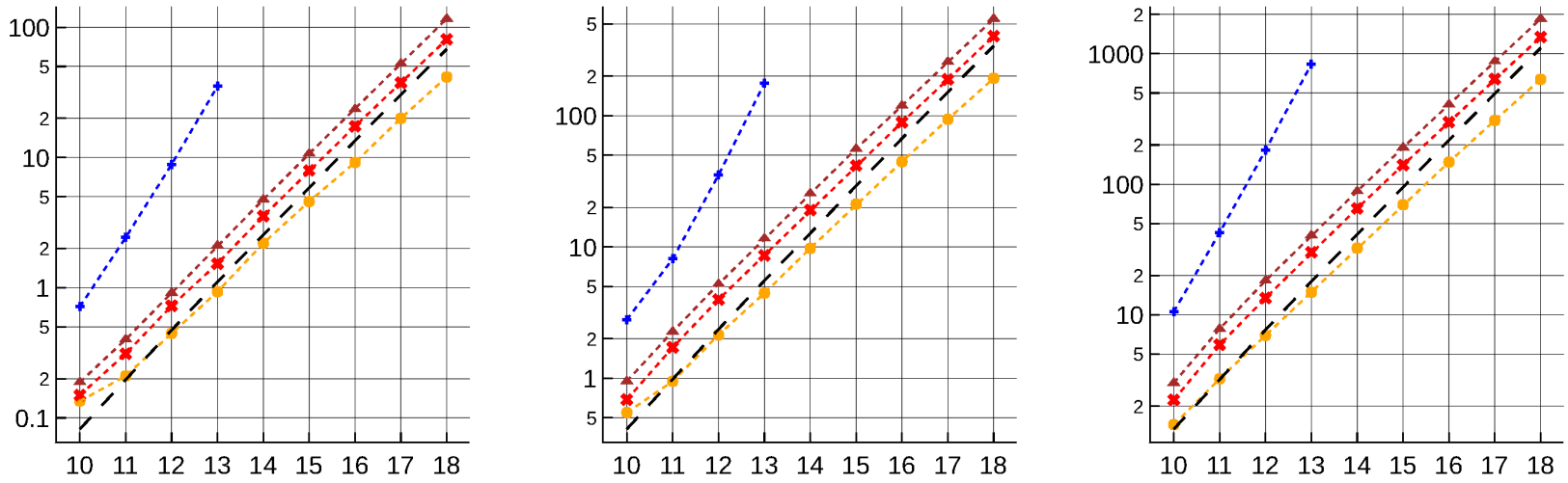



Figure: Time scaling for $\ell_H(\boldsymbol{\theta})$ (left), $\nabla\ell_H(\boldsymbol{\theta})$ (middle), and $H\ell_H(\boldsymbol{\theta})$ (right) on $\log_2$ x-axis; exact times in blue.

**All operations scale quasilinearly.**

# Accuracy of stochastic approximations away from the MLE

| | $n = 2^{10}$ | $n = 2^{11}$ | $n = 2^{12}$ | $n = 2^{13}$ |
|---|---|---|---|---|
| $\varepsilon_{\widehat{\nabla l_H(\boldsymbol{\theta})}}$ | -3.78 | -3.46 | -3.51 | -3.60 |
| $\varepsilon_{\widehat{H l_H(\boldsymbol{\theta})}}$ | -4.22 | -3.89 | -3.71 | -3.79 |

- Away from the MLE, average relative precision (on $\log_{10}$ scale) is pretty good!
- Near the MLE, choosing the right precision metric is less straightforward.

# Accuracy of stochastic approximations at the MLE

| | $n = 2^{10}$ | $n = 2^{11}$ | $n = 2^{12}$ | $n = 2^{13}$ |
|---|---|---|---|---|
| $\varepsilon_{\widehat{\nabla l_H(\boldsymbol{\theta})}}$ | -0.62 | -1.42 | -0.98 | -1.75 |
| $\varepsilon_{\widehat{Hl_H(\boldsymbol{\theta})}}$ | -2.37 | -1.86 | -2.13 | -2.04 |
| $\eta_g$ | -0.92 | -0.93 | -0.73 | -0.96 |
| $\eta_{\mathcal{I}}$ | -1.77 | -1.82 | -1.86 | -2.21 |

Here, we define

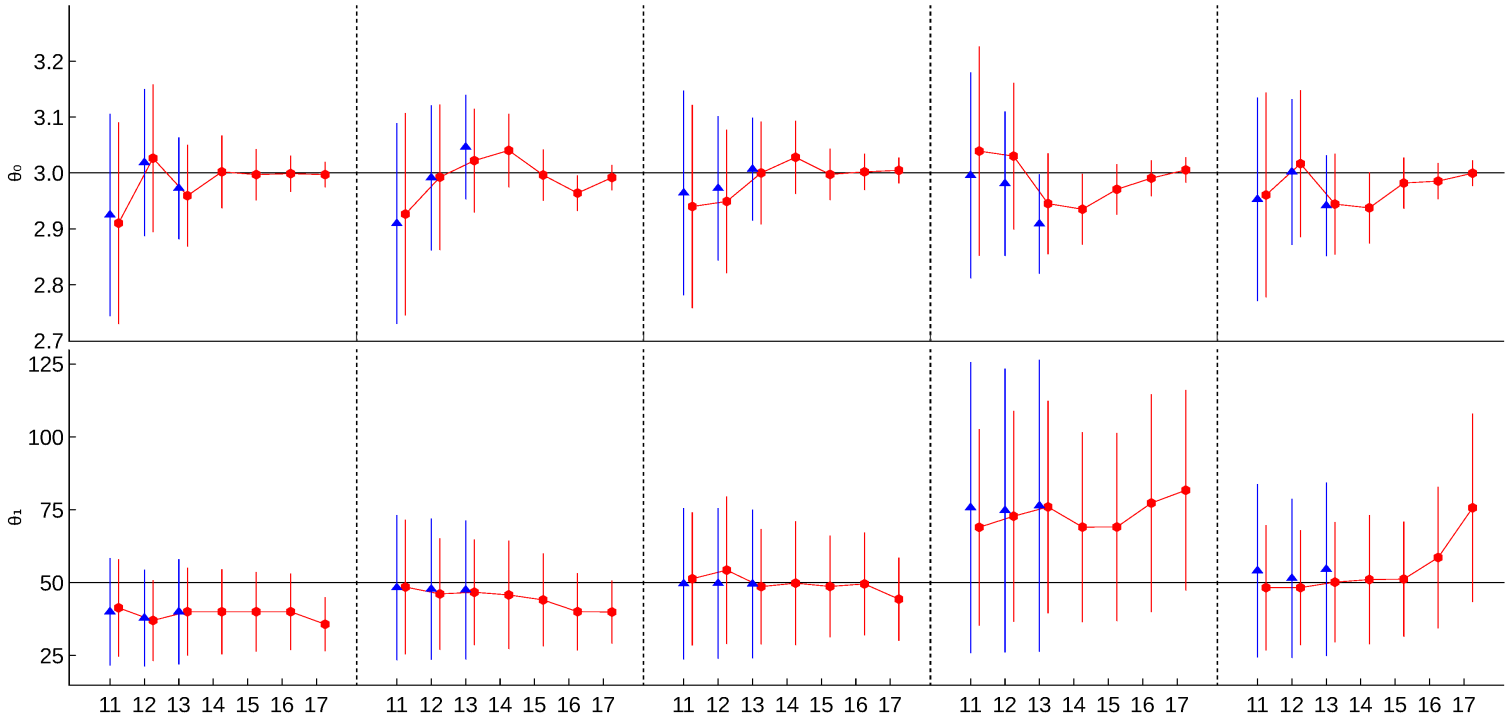
$$\eta_g := ||\widehat{\nabla l_H(\boldsymbol{\theta})} - \nabla l_H(\boldsymbol{\theta})||_{\mathcal{I}(\boldsymbol{\theta})^{-1}}$$

for the gradient and

$$\eta_{\mathcal{I}} := \text{tr}\left\{ \left(\widehat{\mathcal{I}}(\boldsymbol{\theta}) - \mathcal{I}(\boldsymbol{\theta})\right)\left(\mathcal{I}(\boldsymbol{\theta})^{-1} - \widehat{\mathcal{I}}(\boldsymbol{\theta})^{-1}\right) \right\}^{1/2}$$

for the Hessian.

# Accuracy of approximated MLEs




Fitting successively larger samples from the same dataset: approximated point estimates and confidence intervals (red) agree well with exactly computed ones (blue).

# Conclusions and Discussion

# Summary:

- *Differentiable* matrix compression/approximation.
- $\ell_H(\boldsymbol{\theta})$ actually has derivatives, and you can compute them!
- Optimization is flexible and scalable.
- Disregarding rank structure concerns, geometry-, kernel-, and dimension-independent.
- Julia software ready-to-go at `https://bitbucket.org/cgeoga/kernelmatrices.jl.git`

# Some thoughts:

- Somewhere between numerical and model approximation.
- The code will run in any dimension...but should it?
- Prioritizing computability over approximation control.

# The manuscript:



**Statistics > Computation**

## Scalable Gaussian Process Computations Using Hierarchical Matrices

Christopher J. Geoga, Mihai Anitescu, Michael L. Stein

*(Submitted on 9 Aug 2018)*

We present a kernel-independent method that applies hierarchical matrices to the problem of maximum likelihood estimation for Gaussian processes. The proposed approximation provides natural and scalable stochastic estimators for its gradient and Hessian, as well as the expected Fisher information matrix, that are computable in quasilinear $O(n \log^2 n)$ complexity for a large range of models. To accomplish this, we (i) choose a specific hierarchical approximation for covariance matrices that enables the computation of their exact derivatives and (ii) use a stabilized form of the Hutchinson stochastic trace estimator. Since both the observed and expected information matrices can be computed in quasilinear complexity, covariance matrices for MLEs can also be estimated efficiently. After discussing the associated mathematics, we demonstrate the scalability of the method, discuss details of its implementation, and validate that the resulting MLEs and confidence intervals based on the inverse Fisher information matrix faithfully approach those obtained by the exact likelihood.

Subjects: **Computation (stat.CO)**; Numerical Analysis (math.NA); Optimization and Control (math.OC)
Cite as: **arXiv:1808.03215 [stat.CO]**
(or **arXiv:1808.03215v1 [stat.CO]** for this version)

**Submission history**

From: Christopher Geoga [view email]
**[v1]** Thu, 9 Aug 2018 16:04:25 UTC (180 KB)

# The future:

- **Nonstationary models.**
- Strong admissibility?
- Smarter optimization?

Thank you very much for listening!

# References


- Ambikasaran, S. and E. Darve (2013). "An O(nlogn) Fast Direct Solver for Partially Hierarchically Semi-Separable Matrices". In: *Journal of Scientific Computing* 57.3, pp. 477–501.
- Anitescu, M., J. Chen, and L. Wang (2011). "A matrix-free approach for solving the parametric Gaussian process maximum likelihood problem". In: *SIAM J. Sci. Comput.* 34.1, pp. 240–262.
- Bebendorf, M. (2000). "Approximation of boundary element matrices". In: *Numerische Mathematik* 86.4, pp. 565–589.
- Greengard, L. and V. Rokhlin (1987). "A Fast Algorithm for Particle Simulations". In: *J. Comput. Phys.* 73.2, pp. 325–348.
- Hackbusch, W. (1999). "A Sparse Matrix Arithmetic Based on H-Matrices, Part I: Introduction to H-matrices". In: *Computing* 62.2, pp. 89–108.
- Hutchinson, M.F. (1990). "A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines". In: *Communications in Statistics – Simulation and Computation* 19.2, pp. 433–450.
- Rue, H. and L. Held (2005). *Gaussian Markov random fields: theory and applications*. CRC press.
- Stein, M.L. (1999). *Interpolation of Spatial Data: Some Theory for Kriging*. Springer. ISBN: 978-1-4612-7166-6.
- Stein, M.L., J. Chen, and M. Anitescu (2013). "Stochastic approximation of score functions for Gaussian processes". In: *Ann. Appl. Stat.* 7.2, pp. 1162–1191.
- Vecchia, A.V. (1988). "Estimation and model identification for continuous spatial processes". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 50.2, pp. 297–312.
- Williams, C. and M. Seeger (2001). "Using the Nyström Method to Speed Up Kernel Machines". In: *Advances in Neural Information Processing Systems 13*. MIT Press, pp. 682–688.