Sparse Cholesky Factorization

for solving PDEs with Gaussian processes

Yifan Chen

Applied and Computational Mathematics, Caltech

May 15, 2023

The Paper **[Chen, Owhadi, Schäfer 2023]**

"Sparse Cholesky Factorization for Solving Nonlinear PDEs via Gaussian Processes"

Houman Owhadi
Caltech

Florian Schäfer
Georgia Tech

Link: `https://arxiv.org/abs/2304.01294`.

Outline

Outline

Context

Gaussian processes and kernel methods are widely used in scientific computing and scientific machine learning

- Solving PDEs and inverse problems
- Spatial statistics
- Machine learning
- Bayes optimization
- ...

Computational Challenges

Dense kernel matrices, possibly with derivatives

Example:

$$\Theta = \begin{pmatrix} k(\mathbf{x}_\Omega, \mathbf{x}_\Omega) & k(\mathbf{x}_\Omega, \mathbf{x}_{\partial\Omega}) & \Delta_\mathbf{y} k(\mathbf{x}_\Omega, \mathbf{x}_\Omega) \\ k(\mathbf{x}_{\partial\Omega}, \mathbf{x}_\Omega) & k(\mathbf{x}_{\partial\Omega}, \mathbf{x}_{\partial\Omega}) & \Delta_\mathbf{y} k(\mathbf{x}_{\partial\Omega}, \mathbf{x}_\Omega) \\ \Delta_\mathbf{x} k(\mathbf{x}_\Omega, \mathbf{x}_\Omega) & \Delta_\mathbf{x} k(\mathbf{x}_\Omega, \mathbf{x}_{\partial\Omega}) & \Delta_\mathbf{x}\Delta_\mathbf{y} k(\mathbf{x}_\Omega, \mathbf{x}_\Omega) \end{pmatrix}$$

- $k = k(\mathbf{x}, \mathbf{y})$ where $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$
- $\Delta_\mathbf{x}, \Delta_\mathbf{y}$ are Laplacians regarding the 1st and 2nd variables
- $\mathbf{x}_\Omega$: collection of M_Ω points in Ω
- $\mathbf{x}_{\partial\Omega}$: collection of $M_{\partial\Omega}$ points on $\partial\Omega$
- $\Theta \in \mathbb{R}^{N\times N}$ with $N = 2M_\Omega + M_{\partial\Omega}$ in this example

Derivative entries such as $\Delta_\mathbf{x} k$ arise naturally in PDE problems

[Chen, Hosseni, Owhadi, Stuart 2021]

Fast Algorithms

Cubic bottleneck $O(N^3)$: computing with dense Θ matrix

Many approximate methods:

- Nyström approximation, inducing points, random features, covariance tapering, Hierarchical matrices, wavelets based methods ...
- Mostly developed for the case where there is no derivatives

Our goal

Near-linear complexity algorithm when derivative entries exist

Cubic bottleneck $O(N^3)$: computing with dense Θ matrix

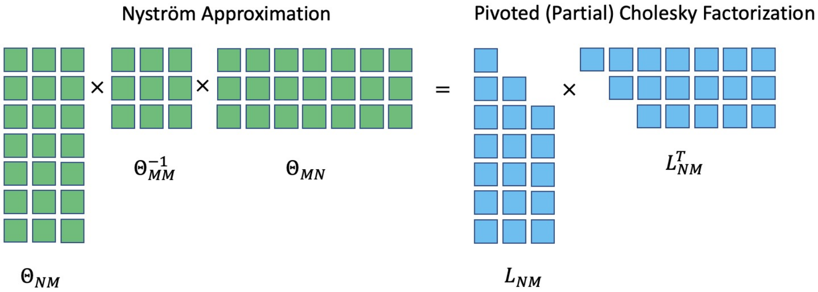
Many approximate methods:

- Nyström approximation, inducing points, random features, covariance tapering, Hierarchical matrices, wavelets based methods ...
- Mostly developed for the case where there is no derivatives

Our goal

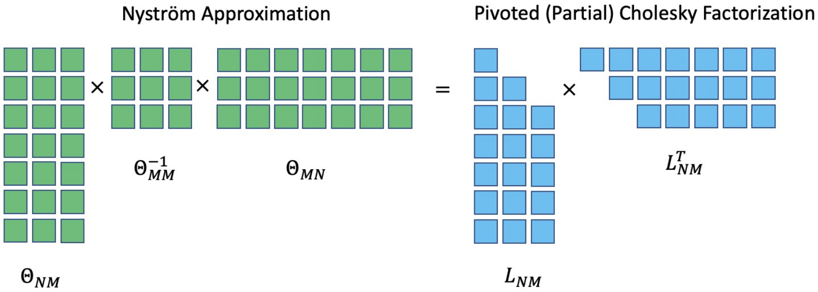
Near-linear complexity algorithm when derivative entries exist

Warm-up: Nyström Approximation



- $\Theta \approx \Theta_{NM}\Theta_{MM}^{-1}\Theta_{MN}$
- Complexity $O(NM^2)$, where M is the number of pivots

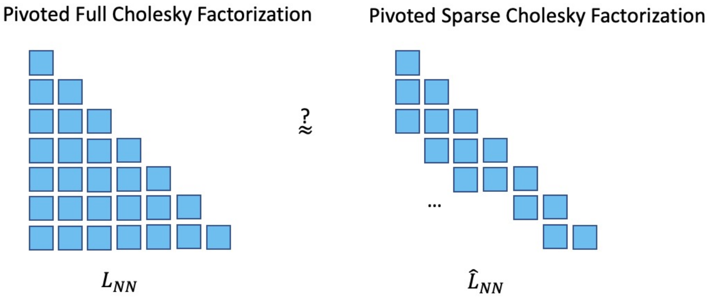
Warm-up: Nyström Approximation



- $\Theta \approx \Theta_{NM}\Theta_{MM}^{-1}\Theta_{MN}$
- Complexity $O(NM^2)$, where M is the number of pivots

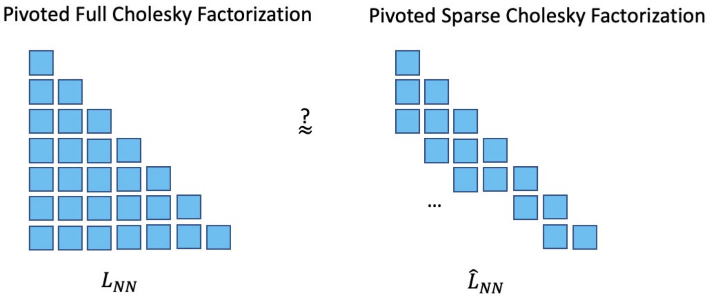
Nevertheless, for high precision in physics problems,
low rank approximation is usually not enough

Full Cholesky Factorization



- Full Cholesky factorization is not affordable: Complexity $O(N^3)$

Full Cholesky Factorization



- Full Cholesky factorization is not affordable:
  Complexity $O(N^3)$

Our focus: Sparse Cholesky factorization

Outline

Sketch of Our Contribution

[Chen, Owhadi, Schäfer 2023]

For Θ with derivative entries, we present a sparse Cholesky factorization algorithm with the state-of-the-art complexity

- $O(N \log^d(N/\epsilon))$ in space; and
- $O(N \log^{2d}(N/\epsilon))$ in time.

The algorithm outputs

- a permutation matrix P_{perm}; and
- a upper triangular matrix U with $O(N \log^d(N/\epsilon))$ nonzeros

such that

$$\|\Theta^{-1} - P_{\mathrm{perm}}^T U U^T P_{\mathrm{perm}}\|_{\mathrm{Fro}} \leq \epsilon$$

where $\|\cdot\|_{\mathrm{Fro}}$ is the Frobenius norm.

Assumptions for all the rigorous results:

- k: Green function of psd differential operator, e.g., $(-\Delta)^s$

How? Probabilistic Interpretation of Cholesky Factorization

Connection between linear algebra and probability

Let $\Theta \in \mathbb{R}^{N\times N}$, and $X \sim \mathcal{N}(0, \Theta)$

- Cholesky factor of the covariance matrix $\Theta = LL^T$

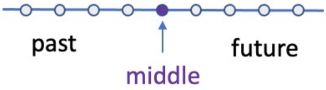
$$\frac{L_{ij}}{L_{jj}} = \frac{\mathrm{Cov}[X_i, X_j | X_{1:j-1}]}{\mathrm{Var}[X_j | X_{1:j-1}]} \qquad (i \geq j)$$

- Cholesky factor of the precision matrix $\Theta^{-1} = UU^T$

$$\frac{U_{ij}}{U_{jj}} = (-1)^{i\neq j} \frac{\mathrm{Cov}[X_i, X_j | X_{1:j-1\setminus\{i\}}]}{\mathrm{Var}[X_j | X_{1:j-1\setminus\{i\}}]} \qquad (i \leq j)$$

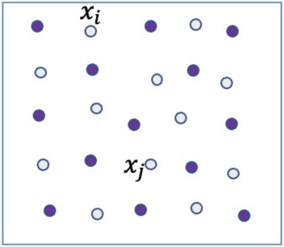
Conditioning, Screening Effects, and Sparsity

Screening effects [Stein 2002]



$k(x, y) = \exp(-|x - y|)$

Cov [past, future | middle] = 0



Matérn's kernel

Cov [fine x_i, fine x_j | coarse] << 1

if x_i and x_j are well separated by coarse points

Sparse Cholesky factors if points ordered from coarse to fine

[Schäfer, Sullivan, Owhadi 2021], [Schäfer, Katzfuss, Owhadi 2021]

How to Order From Coarse to Fine? Maxmin Ordering

Max-min ordering

The next ordered point is the farthest to points selected before

$$\mathbf{x}_k = \mathsf{argmax}_{\mathbf{x}_i} \mathrm{dist}(\mathbf{x}_i, \{\mathbf{x}_j, 1 \leq j < k\})$$

with its lengthscale defined by

$$l_k = \mathrm{dist}(\mathbf{x}_k, \{\mathbf{x}_j, 1 \leq j < k\})$$

- Lead to developments of rigorous sparse Cholesky factorization algorithm for kernel matrices without derivative entries [Schäfer, Sullivan, Owhadi 2021], [Schäfer, Katzfuss, Owhadi 2021]

How about when derivative entries exist?

How to Order From Coarse to Fine? Maxmin Ordering

Max-min ordering

The next ordered point is the farthest to points selected before

$$\mathbf{x}_k = \mathsf{argmax}_{\mathbf{x}_i} \mathrm{dist}(\mathbf{x}_i, \{\mathbf{x}_j, 1 \leq j < k\})$$

with its lengthscale defined by

$$l_k = \mathrm{dist}(\mathbf{x}_k, \{\mathbf{x}_j, 1 \leq j < k\})$$

- Lead to developments of rigorous sparse Cholesky factorization algorithm for kernel matrices without derivative entries
  [Schäfer, Sullivan, Owhadi 2021], [Schäfer, Katzfuss, Owhadi 2021]

How about when derivative entries exist?

How to Order From Coarse to Fine? Maxmin Ordering

Max-min ordering

The next ordered point is the farthest to points selected before

$$\mathbf{x}_k = \text{argmax}_{\mathbf{x}_i} \mathrm{dist}(\mathbf{x}_i, \{\mathbf{x}_j, 1 \leq j < k\})$$

with its lengthscale defined by

$$l_k = \mathrm{dist}(\mathbf{x}_k, \{\mathbf{x}_j, 1 \leq j < k\})$$

- Lead to developments of rigorous sparse Cholesky factorization algorithm for kernel matrices without derivative entries
  [Schäfer, Sullivan, Owhadi 2021], [Schäfer, Katzfuss, Owhadi 2021]

How about when derivative entries exist?

Existence of Sparse Factors

Our new ordering when derivative entries are present

Order the pointwise entries by max-min ordering of the points, then followed with arbitrary order of derivative entries

- Derivative entries treated as *finer scales* than pointwise ones

Theorem [Chen, Owhadi, Schäfer 2023]

Under the above ordering, consider the upper triangular Cholesky factorization $\Theta_{\text{reordered}}^{-1} = U^{\star}U^{\star T}$. Then, for $1 \le i \le j \le N$,

$$|U_{ij}^{\star}| \le C l_j^{\alpha} \exp\left(-\frac{\operatorname{dist}(\mathbf{x}_{P(i)}, \mathbf{x}_{P(j)})}{C l_j}\right)$$

for some generic constant C, α. Here $\mathbf{x}_{P(i)}$ is the physical point corresponding to the ith ordered entry

Existence of Sparse Factors

Our new ordering when derivative entries are present

Order the pointwise entries by max-min ordering of the points, then followed with arbitrary order of derivative entries

- Derivative entries treated as *finer scales* than pointwise ones

Theorem [Chen, Owhadi, Schäfer 2023]

Under the above ordering, consider the upper triangular Cholesky factorization $\Theta_{\text{reordered}}^{-1} = U^{\star}U^{\star T}$. Then, for $1 \leq i \leq j \leq N$,

$$|U_{ij}^{\star}| \leq Cl_j^{\alpha} \exp\left(-\frac{\mathrm{dist}(\mathbf{x}_{P(i)}, \mathbf{x}_{P(j)})}{Cl_j}\right)$$

for some generic constant C, α. Here $\mathbf{x}_{P(i)}$ is the physical point corresponding to the ith ordered entry

Computing Sparse Factors

Entries outside $S_{l,\rho}$ is exponentially small regarding ρ

$$S_{l,\rho} = \{1 \le i \le j \le N : \mathrm{dist}(\mathbf{x}_{P(i)}, \mathbf{x}_{P(j)}) \le \rho l_j\}$$

Algorithm

Using optimization to extract a sparse factor U^ρ

[Schäfer, Katzfuss, Owhadi 2021]

Sparse matrix set: $\mathcal{S}_{l,\rho} = \{A \in \mathbb{R}^{N \times N} : A_{ij} \neq 0 \Rightarrow (i,j) \in S_{l,\rho}\}$

$$U^\rho = \mathrm{argmin}_{U \in \mathcal{S}_{l,\rho}} \mathrm{KL}\left(\mathcal{N}(0, \Theta_{\text{reordered}}) \parallel \mathcal{N}(0, (UU^T)^{-1})\right)$$

- Explicit solution formula for the optimization
- Can be implemented[1] with complexity $O(N\rho^d)$ in space and $O(N\rho^{2d})$ time
- Theory: $\rho = O(\log(N/\epsilon)) \Rightarrow \|\Theta^{-1}_{\text{reordered}} - U^\rho(U^\rho)^T\|_{\text{Fro}} \le \epsilon$

[1] After using the supernode trick to reduce redundant computations

Computing Sparse Factors

Entries outside $S_{l,\rho}$ is exponentially small regarding ρ

$$S_{l,\rho} = \{1 \le i \le j \le N : \mathrm{dist}(\mathbf{x}_{P(i)}, \mathbf{x}_{P(j)}) \le \rho l_j\}$$

Algorithm

Using optimization to extract a sparse factor U^ρ

[Schäfer, Katzfuss, Owhadi 2021]

Sparse matrix set: $\mathcal{S}_{l,\rho} = \{A \in \mathbb{R}^{N\times N} : A_{ij} \neq 0 \Rightarrow (i,j) \in S_{l,\rho}\}$

$$U^\rho = \mathrm{argmin}_{U \in \mathcal{S}_{l,\rho}} \mathrm{KL}\left(\mathcal{N}(0, \Theta_{\text{reordered}}) \parallel \mathcal{N}(0, (UU^T)^{-1})\right)$$

- Explicit solution formula for the optimization
- Can be implemented[1] with complexity $O(N\rho^d)$ in space and $O(N\rho^{2d})$ time
- Theory: $\rho = O(\log(N/\epsilon)) \Rightarrow \|\Theta^{-1}_{\text{reordered}} - U^\rho (U^\rho)^T\|_{\text{Fro}} \le \epsilon$

[1] After using the supernode trick to reduce redundant computations

Computing Sparse Factors

Entries outside $S_{l,\rho}$ is exponentially small regarding ρ

$$S_{l,\rho} = \{1 \le i \le j \le N : \mathrm{dist}(\mathbf{x}_{P(i)}, \mathbf{x}_{P(j)}) \le \rho l_j\}$$

Algorithm

Using optimization to extract a sparse factor U^ρ

[Schäfer, Katzfuss, Owhadi 2021]

Sparse matrix set: $\mathcal{S}_{l,\rho} = \{A \in \mathbb{R}^{N\times N} : A_{ij} \neq 0 \Rightarrow (i,j) \in S_{l,\rho}\}$

$$U^\rho = \mathrm{argmin}_{U \in \mathcal{S}_{l,\rho}} \mathrm{KL}\left(\mathcal{N}(0, \Theta_{\text{reordered}}) \parallel \mathcal{N}(0, (UU^T)^{-1})\right)$$

- Explicit solution formula for the optimization
- Can be implemented[1] with complexity $O(N\rho^d)$ in space and $O(N\rho^{2d})$ time
- Theory: $\rho = O(\log(N/\epsilon)) \Rightarrow \|\Theta^{-1}_{\text{reordered}} - U^\rho (U^\rho)^T\|_{\text{Fro}} \le \epsilon$

[1] After using the supernode trick to reduce redundant computations

Outline

Nonlinear Elliptic Equations

- 2D Example: nonlinear elliptic equation with $\tau(u) = u^3$

$$-\Delta u + \tau(u) = f \quad \text{w/ Dirichlet's boundary condition}$$

- $\Omega = [0,1]^2$. Collocation points uniformly distributed

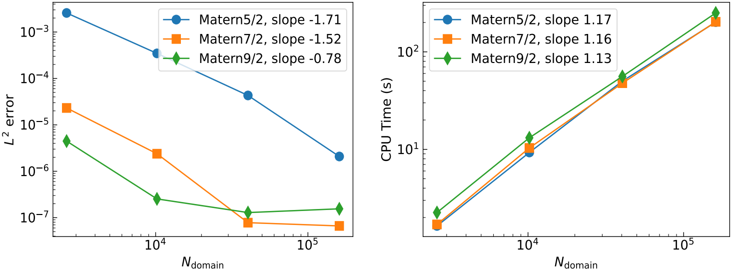
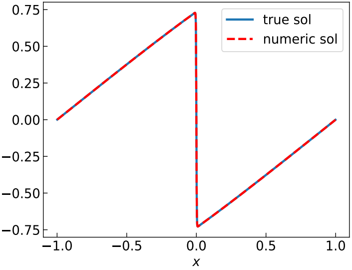


Figure: Run 3 linearization steps with initialization as a zero function. Accuracy floor due to finite ρ

Burgers' Equation

- $\partial_t u + u\partial_x u - 0.001\partial_x^2 u = 0, \quad \forall (x,t) \in (-1,1) \times (0,1]$
- $\Delta t = 0.02, \rho = 4$, solve to $t = 1$



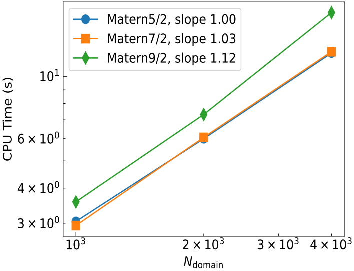


Figure: Run 2 linearization steps at each time step

Monge-Ampère Equation

- Equation: $\det(D^2 u) = f$ in $(0,1)^2$
- Truth $u(\mathbf{x}) = \exp\big(0.5((x_1 - 0.5)^2 + (x_2 - 0,5)^2)\big)$
- Matérn kernel with $\nu = 5/2$, lengthscale 0.3

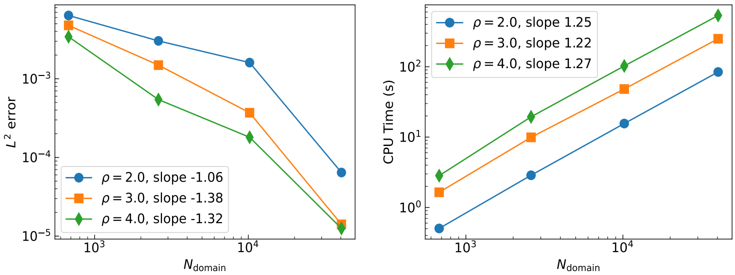


Figure: Run 3 linearization steps with initial guess $1/2\|\mathbf{x}\|^2$. Accuracy floor due to finite ρ

Outline

Summary

Near-linear complexity sparse Cholesky factorization

- Order entries from coarse to fine, with derivative entries treated as finer scales compared to pointwise entries
- This ordering leads to approximately sparse factors
- Computing the inverse Cholesky factors via optimization

Near-linear complexity GP/kernel solver for nonlinear PDEs

- Apply the factorization algorithm into the GP solver
- Each iteration of the algorithm is of near-linear complexity
- Thus a machine learning based near-linear complexity solver for general nonlinear PDEs, assuming the iterations converge (empirically validated)
- Future work: more applications and inverse problems

Thank You

[Chen, Owhadi, Schäfer 2023]

Sparse Cholesky Factorization
for Solving Nonlinear PDEs via Gaussian Processes

Link: `https://arxiv.org/abs/2304.01294`.

Back Up Slides

Explicit Formula

The KL minimization step seeks to find

$$U = \mathrm{argmin}_{\hat{U} \in \mathcal{S}_{P,l,\rho}} \mathrm{KL}\left(\mathcal{N}(0, \Theta) \parallel \mathcal{N}(0, (\hat{U}\hat{U}^T)^{-1})\right) .$$

It turns out that the above problem has an explicit solution

$$U_{s_j,j} = \frac{\Theta^{-1}_{s_j,s_j} \mathbf{e}_{\# s_j}}{\sqrt{\mathbf{e}^T_{\# s_j} \Theta^{-1}_{s_j,s_j} \mathbf{e}_{\# s_j}}} ,$$

where $\mathbf{e}_{\# s_j}$ is a standard basis vector in $\mathbb{R}^{\# s_j}$ with the last entry being 1 and other entries equal 0. Here, $\Theta^{-1}_{s_j,s_j} := (\Theta_{s_j,s_j})^{-1}$.