Sparse Cholesky Factorization by Greedy Conditional Selection

Stephen Huan

https://stephen-huan.github.io/projects/cholesky/

SIAM MDS22

Collaborators

Joe Guinness,
Cornell

Matthias Katzfuß,
Texas A&M

Houman Owhadi,
Caltech

Florian Schäfer,
Gatech

Overview

The problem

Covariance matrices from pairwise kernel function evaluations

i.e. $\Theta_{i,j} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ for points $\{\boldsymbol{x}_i\}_{i=1}^N$ and kernel function K

The problem

Covariance matrices from pairwise kernel function evaluations

i.e. $\Theta_{i,j} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ for points $\{\boldsymbol{x}_i\}_{i=1}^N$ and kernel function K

Kernel trick in machine learning

Statistical inference in Gaussian processes on $\boldsymbol{y} \sim \mathcal{N}(\mathbf{0}, \Theta)$

The problem

Covariance matrices from pairwise kernel function evaluations

i.e. $\Theta_{i,j} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ for points $\{\boldsymbol{x}_i\}_{i=1}^{N}$ and kernel function K

Kernel trick in machine learning

Statistical inference in Gaussian processes on $\boldsymbol{y} \sim \mathcal{N}(\mathbf{0}, \Theta)$

Seek *sparse* Cholesky factor for *dense* covariance matrix

Statistical Cholesky factorization

Factor covariance matrix Θ or precision matrix $Q = \Theta^{-1}$?

$$\Theta_{i,i} = \mathbb{V}\mathrm{ar}[y_i] \qquad\qquad Q^{-1}_{i,i} = \mathbb{V}\mathrm{ar}[y_i \mid y_{k \neq i}]$$

$$\Theta_{i,j} = \mathbb{C}\mathrm{ov}[y_i, y_j] \qquad\qquad \frac{-Q_{i,j}}{\sqrt{Q_{i,i} Q_{j,j}}} = \mathbb{C}\mathrm{orr}[y_i, y_j \mid y_{k \neq i,j}]$$

Cholesky factorization $\Leftrightarrow$ iterative conditioning of process

$$L = \mathrm{chol}(\Theta) \qquad\qquad L = \mathrm{chol}(Q)$$

$$L_{i,j} = \frac{\mathbb{C}\mathrm{ov}[y_i, y_j \mid y_{k < j}]}{\sqrt{\mathbb{V}\mathrm{ar}[y_j \mid y_{k < j}]}} \qquad\qquad -\frac{L_{i,j}}{L_{j,j}} = \frac{\mathbb{C}\mathrm{ov}[y_i, y_j \mid y_{k > j, k \neq i}]}{\mathbb{V}\mathrm{ar}[y_j \mid y_{k > j, k \neq i}]}$$

Statistical Cholesky factorization

Factor covariance matrix Θ or precision matrix $Q = \Theta^{-1}$?

$$\Theta_{i,i} = \mathbb{V}\mathrm{ar}[y_i] \qquad Q^{-1}_{i,i} = \mathbb{V}\mathrm{ar}[y_i \mid y_{k \neq i}]$$

$$\Theta_{i,j} = \mathbb{C}\mathrm{ov}[y_i, y_j] \qquad \frac{-Q_{i,j}}{\sqrt{Q_{i,i} Q_{j,j}}} = \mathbb{C}\mathrm{orr}[y_i, y_j \mid y_{k \neq i,j}]$$

Cholesky factorization $\Leftrightarrow$ iterative conditioning of process

$$L = \mathrm{chol}(\Theta) \qquad L = \mathrm{chol}(Q)$$

$$L_{i,j} = \frac{\mathbb{C}\mathrm{ov}[y_i, y_j \mid y_{k<j}]}{\sqrt{\mathbb{V}\mathrm{ar}[y_j \mid y_{k<j}]}} \qquad -\frac{L_{i,j}}{L_{j,j}} = \frac{\mathbb{C}\mathrm{ov}[y_i, y_j \mid y_{k>j, k \neq i}]}{\mathbb{V}\mathrm{ar}[y_j \mid y_{k>j, k \neq i}]}$$

Conditional (near)-independence $\Leftrightarrow$ (approximate) sparsity

Statistical Cholesky factorization

Factor covariance matrix Θ or precision matrix $Q = \Theta^{-1}$?

$$\Theta_{i,i} = \mathbb{V}\mathrm{ar}[y_i] \qquad Q_{i,i}^{-1} = \mathbb{V}\mathrm{ar}[y_i \mid y_{k \neq i}]$$

$$\Theta_{i,j} = \mathbb{C}\mathrm{ov}[y_i, y_j] \qquad \frac{-Q_{i,j}}{\sqrt{Q_{i,i} Q_{j,j}}} = \mathbb{C}\mathrm{orr}[y_i, y_j \mid y_{k \neq i,j}]$$

Cholesky factorization $\Leftrightarrow$ iterative conditioning of process

$$L = \mathrm{chol}(\Theta) \qquad L = \mathrm{chol}(Q)$$

$$L_{i,j} = \frac{\mathbb{C}\mathrm{ov}[y_i, y_j \mid y_{k<j}]}{\sqrt{\mathbb{V}\mathrm{ar}[y_j \mid y_{k<j}]}} \qquad -\frac{L_{i,j}}{L_{j,j}} = \frac{\mathbb{C}\mathrm{ov}[y_i, y_j \mid y_{k>j, k \neq i}]}{\mathbb{V}\mathrm{ar}[y_j \mid y_{k>j, k \neq i}]}$$

Conditional (near)-independence $\Leftrightarrow$ (approximate) sparsity

Prefer precision matrix to attenuate density

Cholesky factorization recipe

Implied procedure for computing $LL^\top \approx \Theta^{-1}$

1. Pick an ordering on the rows/columns of Θ
2. Select a sparsity pattern lower triangular w.r.t. ordering
3. Compute entries by minimizing objective over all factors

Kullback-Leibler minimization

Compute entries by minimizing Kullback-Leibler divergence

$$L \coloneqq \underset{\hat{L} \in \mathcal{S}}{\operatorname{argmin}} \, \mathbb{D}_{\mathrm{KL}}\Big(\mathcal{N}(\mathbf{0}, \Theta) \, \Big\| \, \mathcal{N}(\mathbf{0}, (\hat{L}\hat{L}^\top)^{-1})\Big)$$

Kullback-Leibler minimization

Compute entries by minimizing Kullback-Leibler divergence

$$L := \operatorname*{argmin}_{\hat{L} \in \mathcal{S}} \mathbb{D}_{\mathrm{KL}}\Big(\mathcal{N}(\mathbf{0}, \Theta) \,\Big\|\, \mathcal{N}(\mathbf{0}, (\hat{L}\hat{L}^\top)^{-1})\Big)$$

Efficient and embarrassingly parallel closed-form solution

$$L_{s_i,i} = \frac{\Theta_{s_i,s_i}^{-1} \boldsymbol{e}_1}{\sqrt{\boldsymbol{e}_1^\top \Theta_{s_i,s_i}^{-1} \boldsymbol{e}_1}}$$

Kullback-Leibler minimization

Compute entries by minimizing Kullback-Leibler divergence

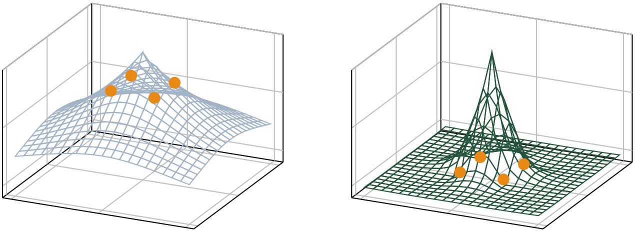
$$L := \underset{\hat{L} \in \mathcal{S}}{\operatorname{argmin}} \, \mathbb{D}_{\mathrm{KL}}\Big(\mathcal{N}(\mathbf{0}, \Theta) \, \Big\| \, \mathcal{N}(\mathbf{0}, (\hat{L}\hat{L}^\top)^{-1})\Big)$$

Efficient and embarrassingly parallel closed-form solution

$$L_{s_i,i} = \frac{\Theta_{s_i,s_i}^{-1} \boldsymbol{e}_1}{\sqrt{\boldsymbol{e}_1^\top \Theta_{s_i,s_i}^{-1} \boldsymbol{e}_1}}$$

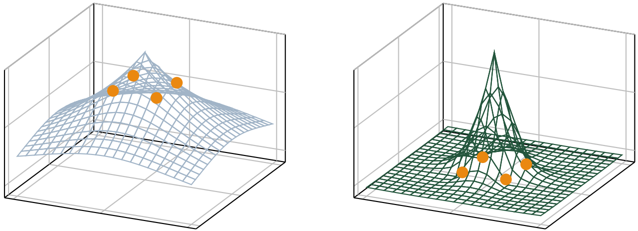
Achieves state of the art ϵ-accuracy in time complexity $\mathcal{O}\left(N \log^{2d}\left(\frac{N}{\epsilon}\right)\right)$ with $\mathcal{O}\left(N \log^{d}\left(\frac{N}{\epsilon}\right)\right)$ nonzero entries [Schäfer, Katzfuss, and Owhadi 2021]

Screening effect



Conditional on points near a point of interest,
far away points are almost independent [Stein 2002]

Screening effect



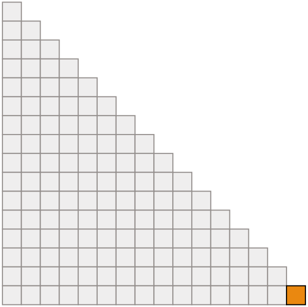
Conditional on points near a point of interest,
far away points are almost independent [Stein 2002]

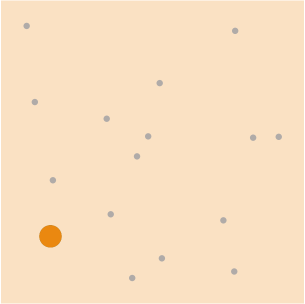
Suggests space-covering ordering and selecting nearby points

Ordering and sparsity pattern

(Reverse) maximin ordering [Guinness 2018] selects the next point x_i with largest distance ℓ_i to points selected before

The ith column selects all points within a radius of $\rho\ell_i$ from x_i

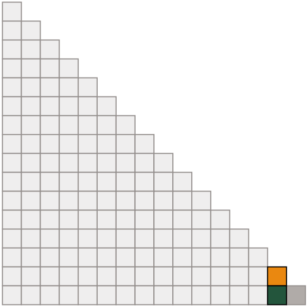


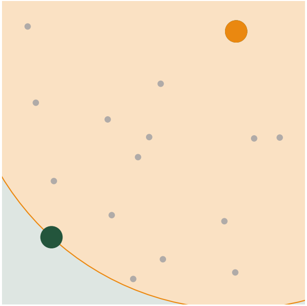


Ordering and sparsity pattern

(Reverse) maximin ordering [Guinness 2018] selects the next point x_i with largest distance ℓ_i to points selected before

The ith column selects all points within a radius of $\rho\ell_i$ from x_i

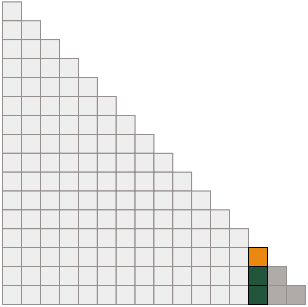


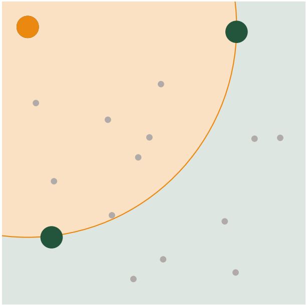


Ordering and sparsity pattern

(Reverse) maximin ordering [Guinness 2018] selects the next point x_i with largest distance ℓ_i to points selected before

The ith column selects all points within a radius of $\rho \ell_i$ from x_i

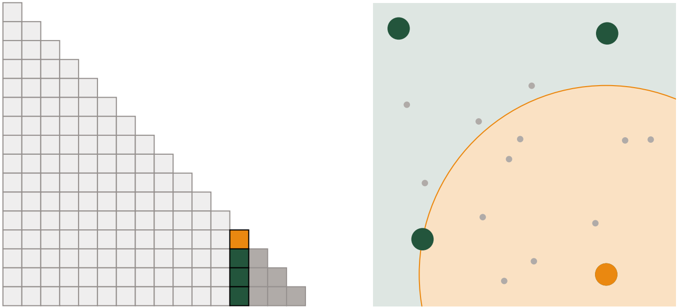




Ordering and sparsity pattern

(Reverse) maximin ordering [Guinness 2018] selects the next point x_i with largest distance ℓ_i to points selected before

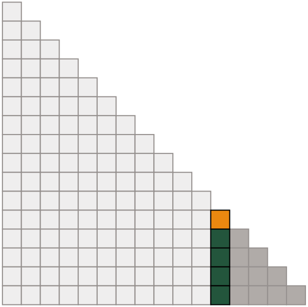
The ith column selects all points within a radius of $\rho\ell_i$ from x_i

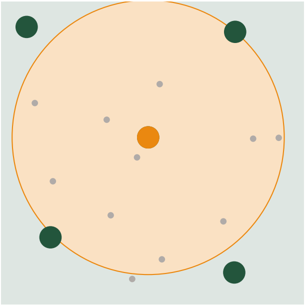


Ordering and sparsity pattern

(Reverse) maximin ordering [Guinness 2018] selects the next point x_i with largest distance ℓ_i to points selected before

The ith column selects all points within a radius of $\rho\ell_i$ from x_i

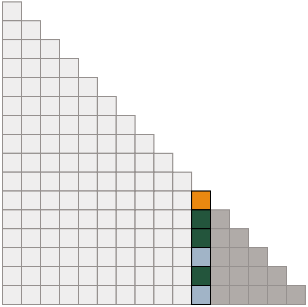


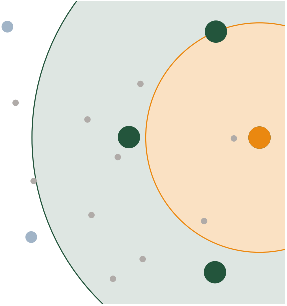


Ordering and sparsity pattern

(Reverse) maximin ordering [Guinness 2018] selects the next point x_i with largest distance ℓ_i to points selected before

The ith column selects all points within a radius of $\rho\ell_i$ from x_i

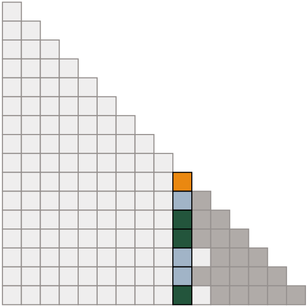


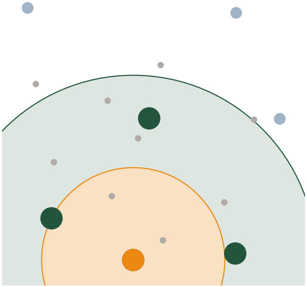


Ordering and sparsity pattern

(Reverse) maximin ordering [Guinness 2018] selects the next point x_i with largest distance ℓ_i to points selected before

The ith column selects all points within a radius of $\rho\ell_i$ from x_i

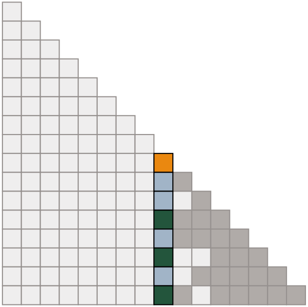


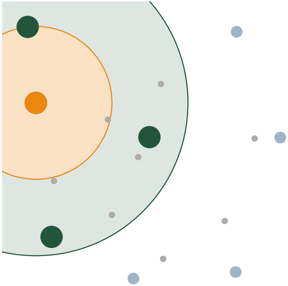


Ordering and sparsity pattern

(Reverse) maximin ordering [Guinness 2018] selects the next point x_i with largest distance ℓ_i to points selected before

The ith column selects all points within a radius of $\rho\ell_i$ from x_i

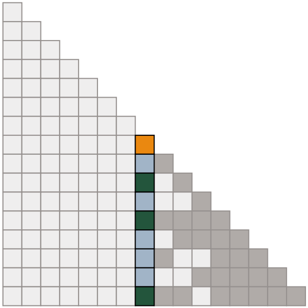


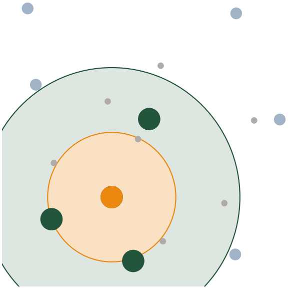


Ordering and sparsity pattern

(Reverse) maximin ordering [Guinness 2018] selects the next point x_i with largest distance ℓ_i to points selected before

The ith column selects all points within a radius of $\rho\ell_i$ from x_i

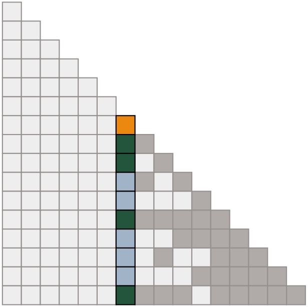


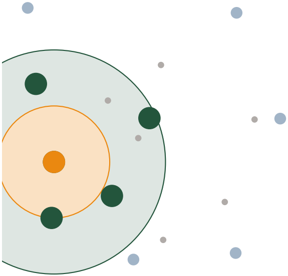


Ordering and sparsity pattern

(Reverse) maximin ordering [Guinness 2018] selects the next point x_i with largest distance ℓ_i to points selected before

The ith column selects all points within a radius of $\rho \ell_i$ from x_i

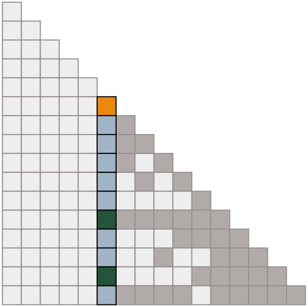


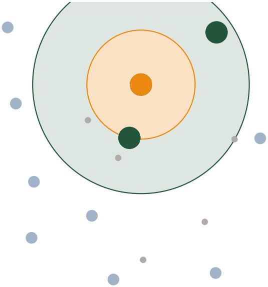


Ordering and sparsity pattern

(Reverse) maximin ordering [Guinness 2018] selects the next point x_i with largest distance ℓ_i to points selected before

The ith column selects all points within a radius of $\rho\ell_i$ from x_i

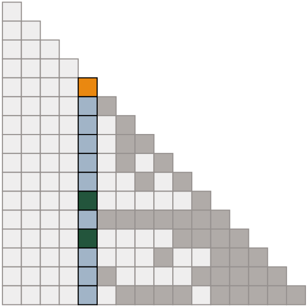


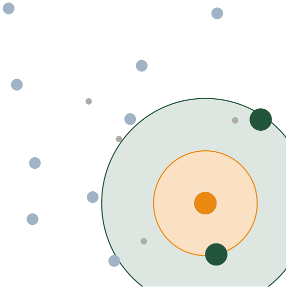


Ordering and sparsity pattern

(Reverse) maximin ordering [Guinness 2018] selects the next point x_i with largest distance ℓ_i to points selected before

The ith column selects all points within a radius of $\rho\ell_i$ from x_i

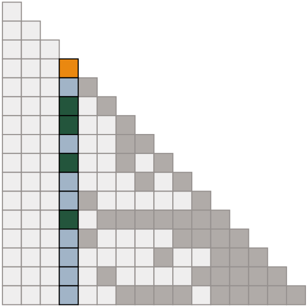


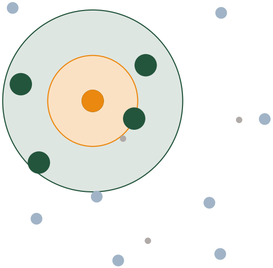


Ordering and sparsity pattern

(Reverse) maximin ordering [Guinness 2018] selects the next point x_i with largest distance ℓ_i to points selected before

The ith column selects all points within a radius of $\rho \ell_i$ from x_i

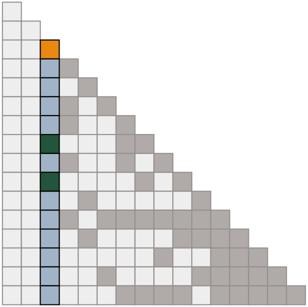


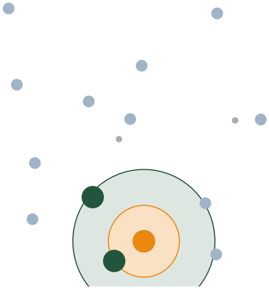


Ordering and sparsity pattern

(Reverse) maximin ordering [Guinness 2018] selects the next point x_i with largest distance ℓ_i to points selected before

The ith column selects all points within a radius of $\rho\ell_i$ from x_i

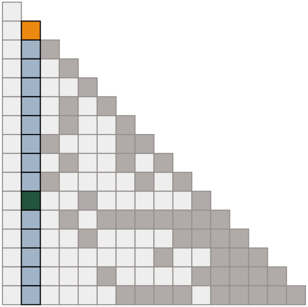


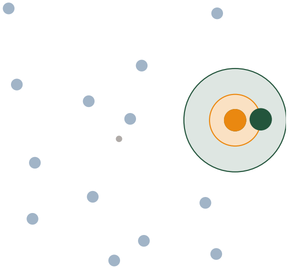


Ordering and sparsity pattern

(Reverse) maximin ordering [Guinness 2018] selects the next point x_i with largest distance ℓ_i to points selected before

The ith column selects all points within a radius of $\rho\ell_i$ from x_i

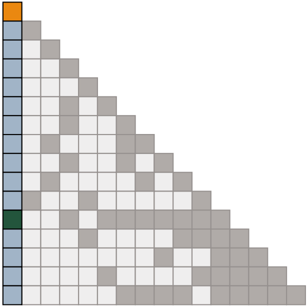


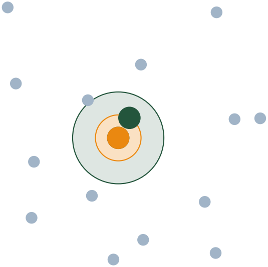


Ordering and sparsity pattern

(Reverse) maximin ordering [Guinness 2018] selects the next point $\boldsymbol{x}_i$ with largest distance ℓ_i to points selected before

The ith column selects all points within a radius of $\rho\ell_i$ from $\boldsymbol{x}_i$





This work: KL-minimization, revisited

Plug optimal L back into the KL divergence

$$\mathbb{D}_{\mathrm{KL}}\Big(\Theta \,\Big\|\, (LL^\top)^{-1}\Big) = \sum_{i=1}^{N} \left[\log\left(\Theta_{i,i|s_i \setminus \{i\}}\right) - \log\left(\Theta_{i,i|i+1:}\right)\right]$$

This work: KL-minimization, revisited

Plug optimal L back into the KL divergence

$$\mathbb{D}_{\mathrm{KL}}\left(\Theta \,\middle\|\, (LL^\top)^{-1}\right) = \sum_{i=1}^{N} \left[\log\left(\Theta_{i,i|s_i \setminus \{i\}}\right) - \log\left(\Theta_{i,i|i+1:}\right)\right]$$

KL $\Leftrightarrow$ accumulated error over independent regression problems

This work: KL-minimization, revisited

Plug optimal L back into the KL divergence

$$\mathbb{D}_{\mathrm{KL}}\left(\Theta \,\middle\|\, (LL^\top)^{-1}\right) = \sum_{i=1}^{N} \left[\log\left(\Theta_{i,i|s_i \setminus \{i\}}\right) - \log\left(\Theta_{i,i|i+1:}\right)\right]$$

KL $\Leftrightarrow$ accumulated error over independent regression problems

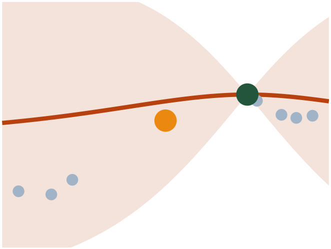
Goal: minimize posterior variance of ith prediction point by selecting training points s_i *most informative* to that point

Variance $\Leftrightarrow$ mutual information $\Leftrightarrow$ mean squared error

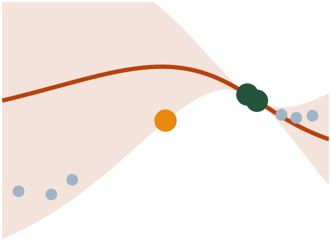
Conditional k-nearest neighbors

Sparse Gaussian process regression, experimental design, active set, etc.

Naive: select k closest points



Conditional k-nearest neighbors



Sparse Gaussian process regression,
experimental design, active set, etc.

Naive: select k closest points

Chooses redundant information

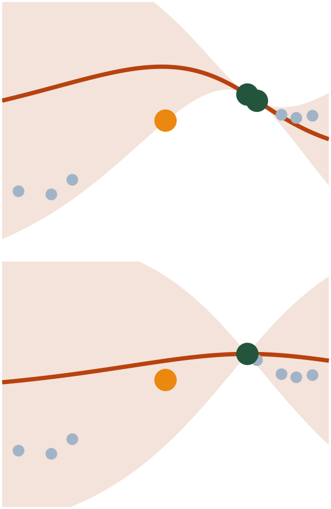
Conditional k-nearest neighbors

Sparse Gaussian process regression, experimental design, active set, etc.

Naive: select k closest points

Chooses redundant information

Maximize *mutual information*!



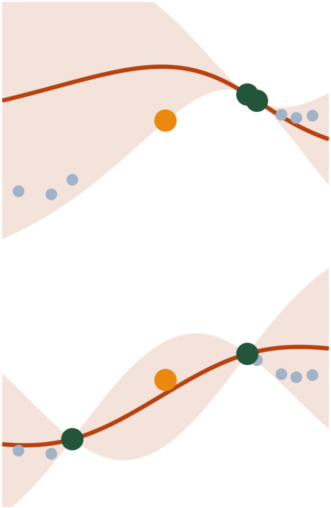
Conditional k-nearest neighbors

Sparse Gaussian process regression, experimental design, active set, etc.

Naive: select k closest points

Chooses redundant information

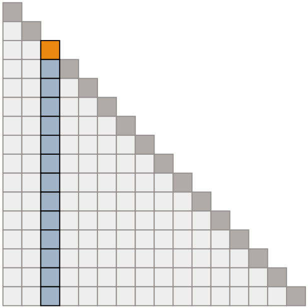
Maximize *mutual information*!

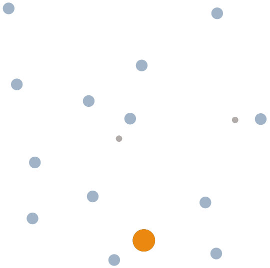


Cholesky factorization by greedy selection

Identify target point as the diagonal entry, candidates are below it, and add selected entries to the sparsity pattern

In practice, restrict candidate set to nearest neighbors, e.g.

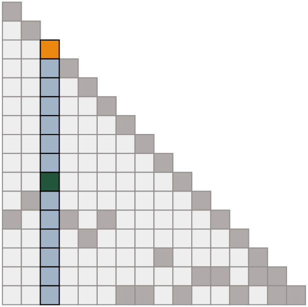


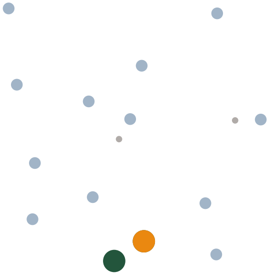


Cholesky factorization by greedy selection

Identify target point as the diagonal entry, candidates are below it, and add selected entries to the sparsity pattern

In practice, restrict candidate set to nearest neighbors, e.g.

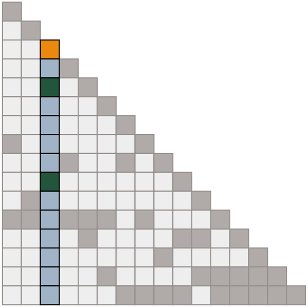


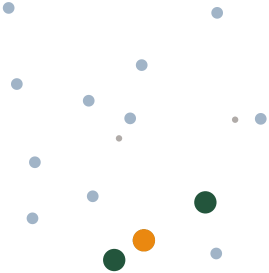


Cholesky factorization by greedy selection

Identify target point as the diagonal entry, candidates are below it, and add selected entries to the sparsity pattern

In practice, restrict candidate set to nearest neighbors, e.g.

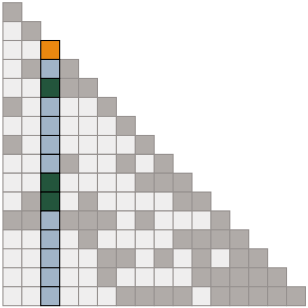


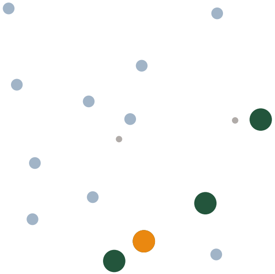


Cholesky factorization by greedy selection

Identify target point as the diagonal entry, candidates are below it, and add selected entries to the sparsity pattern

In practice, restrict candidate set to nearest neighbors, e.g.

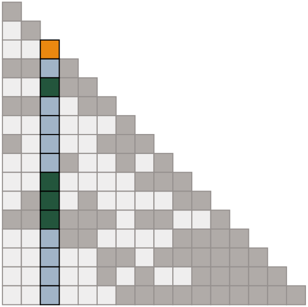


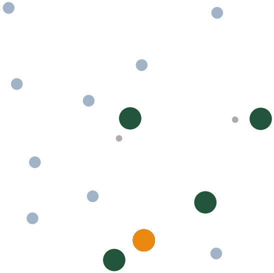


Cholesky factorization by greedy selection

Identify target point as the diagonal entry, candidates are below it, and add selected entries to the sparsity pattern

In practice, restrict candidate set to nearest neighbors, e.g.

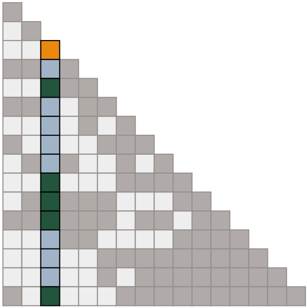


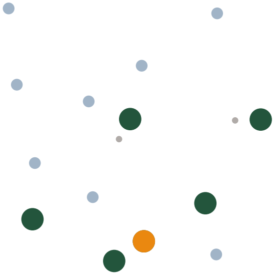


Cholesky factorization by greedy selection

Identify target point as the diagonal entry, candidates are below it, and add selected entries to the sparsity pattern

In practice, restrict candidate set to nearest neighbors, e.g.

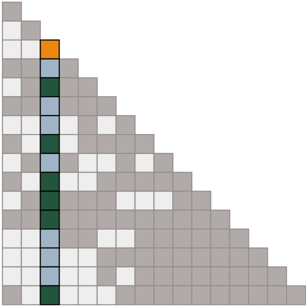


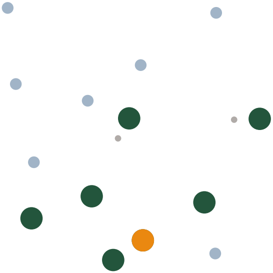


Cholesky factorization by greedy selection

Identify target point as the diagonal entry, candidates are below it, and add selected entries to the sparsity pattern

In practice, restrict candidate set to nearest neighbors, e.g.

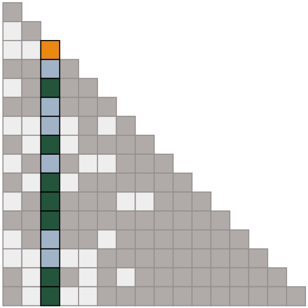


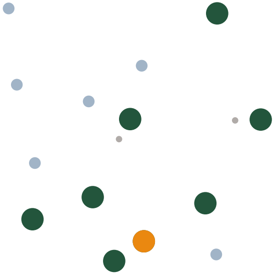


Cholesky factorization by greedy selection

Identify target point as the diagonal entry, candidates are below it, and add selected entries to the sparsity pattern

In practice, restrict candidate set to nearest neighbors, e.g.

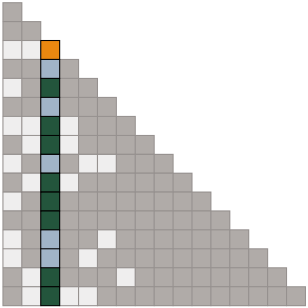


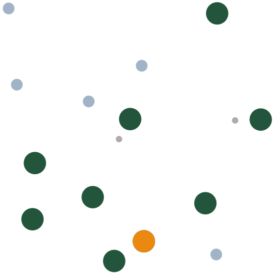


Cholesky factorization by greedy selection

Identify target point as the diagonal entry, candidates are below it, and add selected entries to the sparsity pattern

In practice, restrict candidate set to nearest neighbors, e.g.

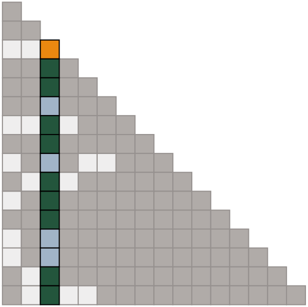


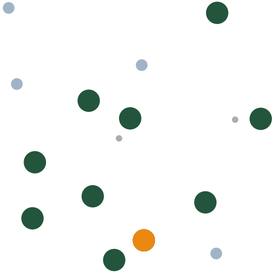


Cholesky factorization by greedy selection

Identify target point as the diagonal entry, candidates are below it, and add selected entries to the sparsity pattern

In practice, restrict candidate set to nearest neighbors, e.g.

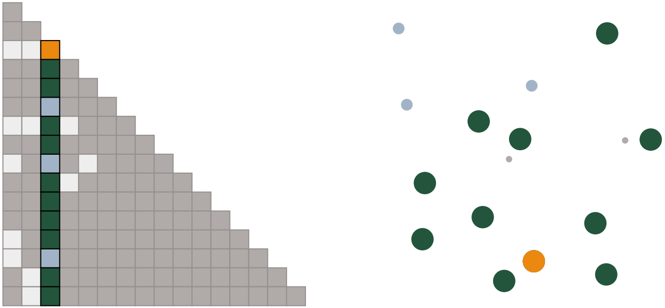




Cholesky factorization by greedy selection

Identify target point as the diagonal entry, candidates are below it, and add selected entries to the sparsity pattern

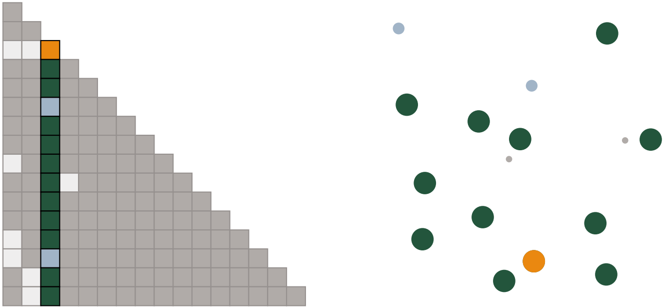
In practice, restrict candidate set to nearest neighbors, e.g.



Cholesky factorization by greedy selection

Identify target point as the diagonal entry, candidates are below it, and add selected entries to the sparsity pattern

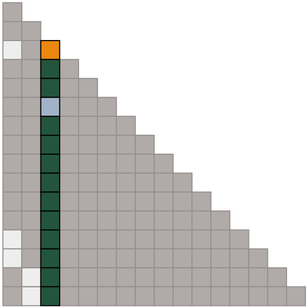
In practice, restrict candidate set to nearest neighbors, e.g.

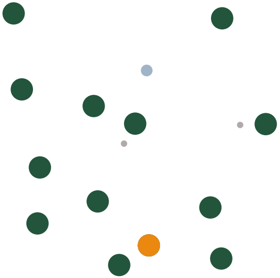


Cholesky factorization by greedy selection

Identify target point as the diagonal entry, candidates are below it, and add selected entries to the sparsity pattern

In practice, restrict candidate set to nearest neighbors, e.g.

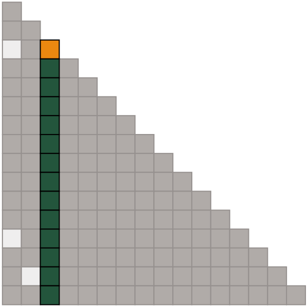


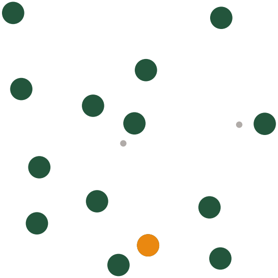


Cholesky factorization by greedy selection

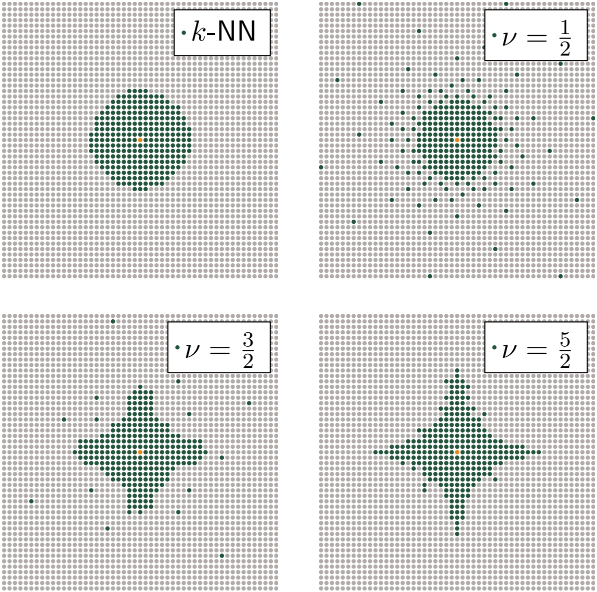
Identify target point as the diagonal entry, candidates are below it, and add selected entries to the sparsity pattern

In practice, restrict candidate set to nearest neighbors, e.g.





Conditional selection



Greedy conditional selection

Intractable to search over $\binom{N}{s}$ subsets, use greedy instead

Greedy conditional selection

Intractable to search over $\binom{N}{s}$ subsets, use greedy instead

Direct computation is $\mathcal{O}(Ns^4)$ to select s points out of N

Greedy conditional selection

Intractable to search over $\binom{N}{s}$ subsets, use greedy instead

Direct computation is $\mathcal{O}(Ns^4)$ to select s points out of N

Maintain partial Cholesky factor for $\mathcal{O}(Ns^2)$

Fast conditional selection

Selecting candidate k is rank-one downdate to covariance Θ

$$\Theta_{:,:|I,k} = \Theta_{:,:|I} - \boldsymbol{u}\boldsymbol{u}^\top \qquad \boldsymbol{u} = \frac{\Theta_{:,k|I}}{\sqrt{\Theta_{k,k|I}}}$$

Fast conditional selection

Selecting candidate k is rank-one downdate to covariance Θ

$$\Theta_{:,:|I,k} = \Theta_{:,:|I} - \boldsymbol{u}\boldsymbol{u}^\top \qquad \boldsymbol{u} = \frac{\Theta_{:,k|I}}{\sqrt{\Theta_{k,k|I}}}$$

Corresponding decrease in posterior variance is

$$u_{\mathsf{Pr}}^2 = \frac{\mathbb{C}\mathrm{ov}[y_{\mathsf{Pr}}, y_k \mid I]^2}{\mathbb{V}\mathrm{ar}[y_k \mid I]} = \mathbb{V}\mathrm{ar}[y_{\mathsf{Pr}} \mid I]\,\mathbb{C}\mathrm{orr}[y_{\mathsf{Pr}}, y_k \mid I]^2$$

Fast conditional selection

Selecting candidate k is rank-one downdate to covariance Θ

$$\Theta_{:,:|I,k} = \Theta_{:,:|I} - \boldsymbol{u}\boldsymbol{u}^{\top} \qquad \boldsymbol{u} = \frac{\Theta_{:,k|I}}{\sqrt{\Theta_{k,k|I}}}$$

Corresponding decrease in posterior variance is

$$u_{\mathsf{Pr}}^2 = \frac{\mathbb{C}\mathrm{ov}[y_{\mathsf{Pr}}, y_k \mid I]^2}{\mathbb{V}\mathrm{ar}[y_k \mid I]} = \mathbb{V}\mathrm{ar}[y_{\mathsf{Pr}} \mid I]\, \mathbb{C}\mathrm{orr}[y_{\mathsf{Pr}}, y_k \mid I]^2$$

Compute $\boldsymbol{u}$ as next column of (partial) Cholesky factor

Fast conditional selection

Selecting candidate k is rank-one downdate to covariance Θ

$$\Theta_{:,:|I,k} = \Theta_{:,:|I} - \boldsymbol{u}\boldsymbol{u}^\top \qquad \boldsymbol{u} = \frac{\Theta_{:,k|I}}{\sqrt{\Theta_{k,k|I}}}$$

Corresponding decrease in posterior variance is

$$u_{\mathsf{Pr}}^2 = \frac{\mathbb{C}\mathrm{ov}[y_{\mathsf{Pr}}, y_k \mid I]^2}{\mathbb{V}\mathrm{ar}[y_k \mid I]} = \mathbb{V}\mathrm{ar}[y_{\mathsf{Pr}} \mid I]\,\mathbb{C}\mathrm{orr}[y_{\mathsf{Pr}}, y_k \mid I]^2$$

Compute $\boldsymbol{u}$ as next column of (partial) Cholesky factor

Replace $\mathcal{O}(N^2)$ update with $\mathcal{O}(Ns)$ by “left-looking”

$$L_{:,i} \leftarrow \Theta_{:,k} - L_{:,:i-1} L_{k,:i-1}^\top$$

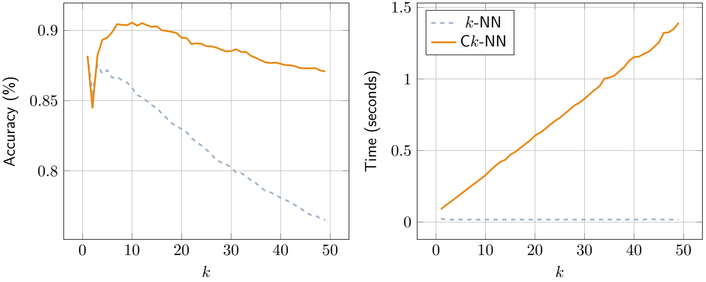
$$L_{:,i} \leftarrow \frac{L_{:,i}}{\sqrt{L_{k,i}}}$$

k-nearest neighbors

Image classification by mode label of k-"nearest" neighbors

MNIST database of handwritten digits [Lecun et al. 1998]

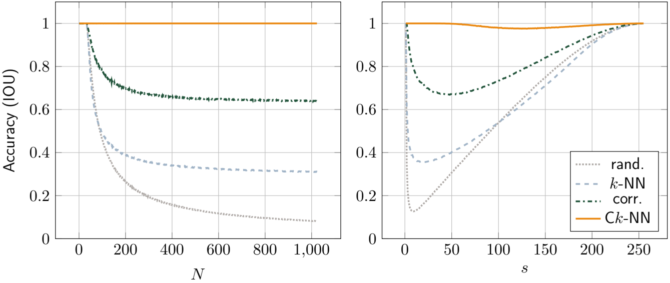
Matérn kernel with smoothness $\nu = \frac{3}{2}$ and length scale $\ell = 2^{10}$



Recovery of sparse factors

Randomly generate *a priori* sparse Cholesky factor L

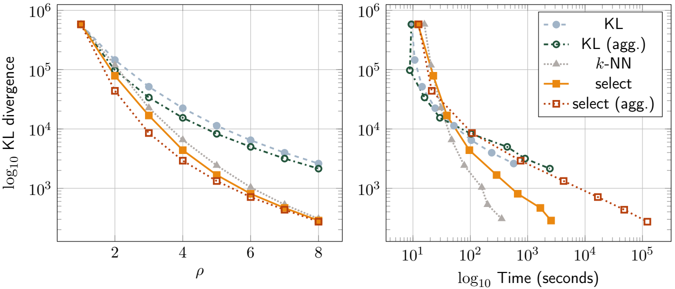
Attempt to recover L given covariance matrix $\Theta = LL^\top$



Cholesky factorization

Randomly sample $N = 2^{16}$ points uniformly from $[0, 1]^3$

Matérn kernel with smoothness $\nu = \frac{5}{2}$ and length scale $\ell = 1$



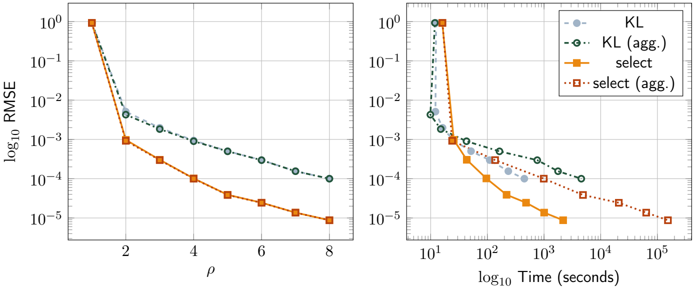
Gaussian process regression

Randomly sample 2^{16} points uniformly from $[0, 1]^3$

Randomly partition into 90% training and 10% prediction

Matérn kernel with smoothness $\nu = \frac{5}{2}$ and length scale $\ell = 1$

Draw 10^3 realizations from the resulting Gaussian process



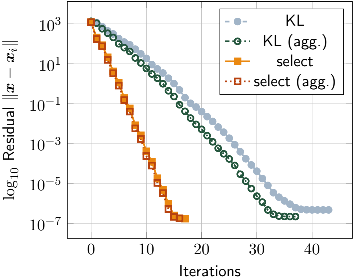
Preconditioning the conjugate gradient

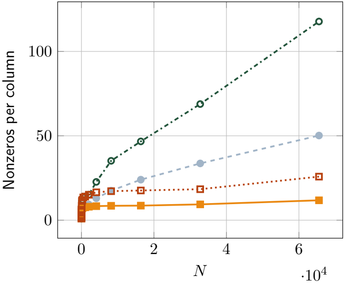
Randomly sample N points uniformly from $[0,1]^3$

Matérn kernel with smoothness $\nu = \frac{1}{2}$ and length scale $\ell = 1$

First sample solution $\boldsymbol{x} \sim \mathcal{N}(\mathbf{0}, \mathrm{Id}_N)$ then compute $\boldsymbol{y} = \Theta \boldsymbol{x}$

Run conjugate gradient with preconditioner L





Summary

Sparse Cholesky factorization of *dense* kernel matrices from approximate conditional independence in Gaussian processes

Previous work exploits screening effect for ordering and sparsity

Replace pure geometry with information-theoretic criteria

More accurate factors at the same sparsity

Conditional selection is computationally efficient

Thank You!

Guinness, Joseph (Oct. 2018). "Permutation and Grouping Methods for Sharpening Gaussian Process Approximations". In: *Technometrics* 60.4, pp. 415–429. ISSN: 0040-1706, 1537-2723. DOI: 10.1080/00401706.2018.1437476. arXiv: 1609.05372 [stat].

Lecun, Y. et al. (Nov. 1998). "Gradient-Based Learning Applied to Document Recognition". In: *Proceedings of the IEEE* 86.11, pp. 2278–2324. ISSN: 1558-2256. DOI: 10.1109/5.726791.

Schäfer, Florian, Matthias Katzfuss, and Houman Owhadi (Oct. 2021). "Sparse Cholesky Factorization by Kullback-Leibler Minimization". In: *arXiv:2004.14455 [cs, math, stat]*. arXiv: 2004.14455 [cs, math, stat].

Stein, Michael L. (Feb. 2002). "The Screening Effect in Kriging". In: *The Annals of Statistics* 30.1, pp. 298–323. ISSN: 0090-5364, 2168-8966. DOI: 10.1214/aos/1015362194.

Mutual information objective

Define *mutual information* or *information gain*

$$\mathbb{I}[\boldsymbol{y}_{\mathsf{Pr}}; \boldsymbol{y}_{\mathsf{Tr}}] = \mathbb{H}[\boldsymbol{y}_{\mathsf{Pr}}] - \mathbb{H}[\boldsymbol{y}_{\mathsf{Pr}} \mid \boldsymbol{y}_{\mathsf{Tr}}]$$

Entropy increasing with log determinant of covariance

Information-theoretic EV-VE identity

$$\begin{aligned} \mathbb{H}[\boldsymbol{y}_{\mathsf{Pr}}] &= \mathbb{H}[\boldsymbol{y}_{\mathsf{Pr}} \mid \boldsymbol{y}_{\mathsf{Tr}}] + \mathbb{I}[\boldsymbol{y}_{\mathsf{Pr}}; \boldsymbol{y}_{\mathsf{Tr}}] \\ \mathbb{V}\mathrm{ar}[\boldsymbol{y}_{\mathsf{Pr}}] &= \mathbb{E}[\mathbb{V}\mathrm{ar}[\boldsymbol{y}_{\mathsf{Pr}} \mid \boldsymbol{y}_{\mathsf{Tr}}]] + \mathbb{V}\mathrm{ar}[\mathbb{E}[\boldsymbol{y}_{\mathsf{Pr}} \mid \boldsymbol{y}_{\mathsf{Tr}}]] \end{aligned}$$

Orthogonal matching pursuit

Conditional selection can be seen as orthogonal matching pursuit in covariance rather than feature space

$$\Theta = F^\top F$$

where F's columns F_i are vectors in feature space and

$$\Theta_{i,j} = \langle F_i, F_j \rangle$$

Suppose F has QR factorization

$$F = QR$$

for Q orthonormal and R upper triangular. Then

$$\begin{aligned} \Theta &= F^\top F = (QR)^\top (QR) \\ &= R^\top Q^\top Q R \\ &= R^\top R \end{aligned}$$

so $R^\top$ is a lower triangular Cholesky factor of Θ.

Multiple prediction points

Select candidate for *multiple* prediction points jointly

Try to take advantage of "two birds with one stone"

Flipped objective allows efficient algorithm by single selection

$$\operatorname{logdet}(\Theta_{\mathsf{Pr},\mathsf{Pr}|I,k}) - \operatorname{logdet}(\Theta_{\mathsf{Pr},\mathsf{Pr}|I}) = \log(\Theta_{k,k|I,\mathsf{Pr}}) - \log(\Theta_{k,k|I})$$

$\mathcal{O}(Ns^2 + Nm^2 + m^3)$ to select s points out of N candidates for m targets, essentially m times faster than single selection

Partial selection

In aggregated (supernodal) Cholesky factorization, "partial" addition of candidates if candidate is between grouped targets

Conditional structure of partially conditioned covariance matrix

$$\mathbb{C}\mathrm{ov}[\boldsymbol{y}_{\|k}] = \begin{pmatrix} L_{:p} L_{:p}^{\top} & L_{:p} {L'}_{p+1:}^{\top} \\ {L'}_{p+1:} L_{:p}^{\top} & {L'}_{p+1:} {L'}_{p+1:}^{\top} \end{pmatrix} = \begin{pmatrix} L_{:p} \\ {L'}_{p+1:} \end{pmatrix} \begin{pmatrix} L_{:p} \\ {L'}_{p+1:} \end{pmatrix}^{\top}$$

Efficient inductive algorithm matches complexity of multiple-target selection algorithm using rank-one downdating

$$\begin{aligned} \Theta_{i,i|:i-1} &= L_{i,i}^2 \\ \Theta_{j,i|:i-1} &= L_{j,i} \cdot L_{i,i} \\ \Theta_{i,i|:i-1,j} &= \Theta_{i,i|:i-1} - \Theta_{j,i|:i-1}^2 / \Theta_{j,j|:i-1} \\ \Theta_{j,j|:i-1,i} &= \Theta_{j,j|:i-1} - \Theta_{j,i|:i-1}^2 / \Theta_{i,i|:i-1} = \Theta_{j,j|:i} \end{aligned}$$

Allocating nonzeros by global selection

It matters how many nonzeros each columns receives, especially for inhomogeneous geometries

Distributing evenly maximizes computational efficiency

To maximize accuracy, maintain *global* priority queue that determines both the next candidate to select and its column

Priority queue implemented as array-backed binary heap, e.g.