Sparse Cholesky Factorization by Greedy Conditional Selection

Stephen Huan and Florian Schäfer

Georgia Institute of Technology

The problem: Gaussian process regression

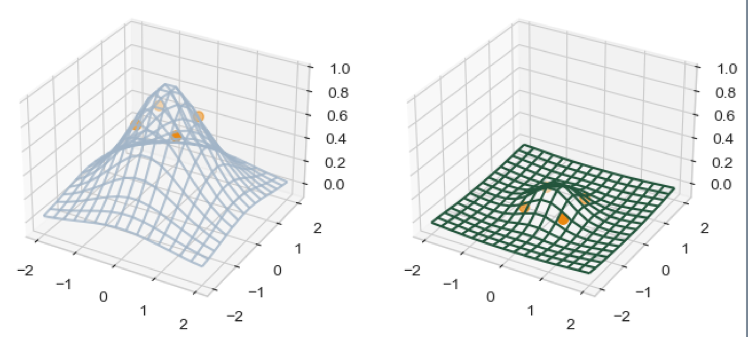
Given measurements $\boldsymbol{y}_{\text{Tr}}$ at N points X_{Tr}, we wish to estimate unseen data $\boldsymbol{y}_{\text{Pr}}$ at X_{Pr}. Estimation of $\boldsymbol{y}_{\text{Pr}}$ can be done as conditioning on $\boldsymbol{y}_{\text{Tr}}$:

$$\mathrm{E}[\boldsymbol{y}_{\text{Pr}} \mid \boldsymbol{y}_{\text{Tr}}] = \boldsymbol{\mu}_{\text{Pr}} + \Theta_{\text{Pr},\text{Tr}} \Theta_{\text{Tr},\text{Tr}}^{-1} (\boldsymbol{y}_{\text{Tr}} - \boldsymbol{\mu}_{\text{Tr}})$$

$$\Theta_{\text{Pr},\text{Pr} \mid \text{Tr}} := \mathrm{Cov}[\boldsymbol{y}_{\text{Pr}} \mid \boldsymbol{y}_{\text{Tr}}] = \Theta_{\text{Pr},\text{Pr}} - \Theta_{\text{Pr},\text{Tr}} \Theta_{\text{Tr},\text{Tr}}^{-1} \Theta_{\text{Tr},\text{Pr}}$$

Cubic bottleneck and the screening effect

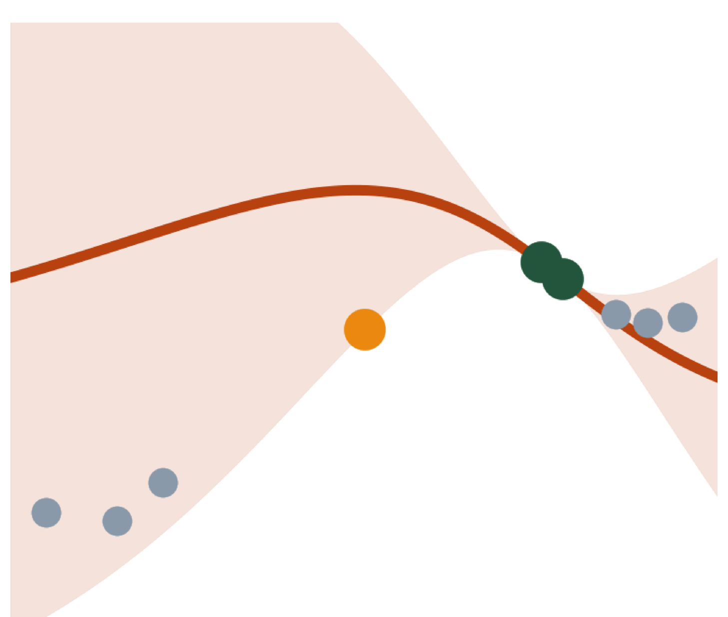
Computing conditional distribution has computational cost $\mathcal{O}(N^3)$, often infeasible! Use "screening": conditional on nearby points, far points have little covariance.



k-th nearest neighbors?

The screening effect suggests that one should simply pick the k closest points, recovering the k-th nearest neighbors (k-NN) algorithm.

Here, the blue points are the candidates, the orange point is the unknown point, and the green points are the k selected points (in this example, $k = 2$).

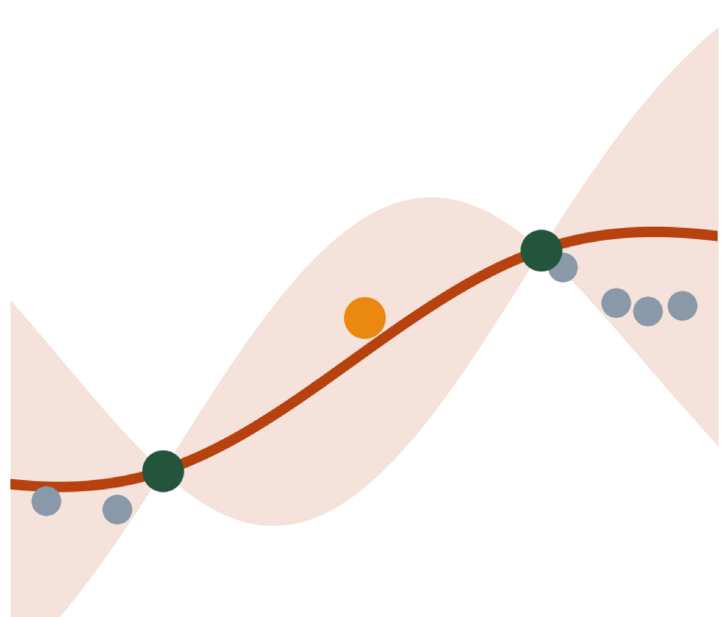


k-NN is myopic, account for conditioning!

Algorithm: [conditional k-th nearest neighbors (Ck-NN)]:

Selecting the closest point every iteration leads to redundancy.

Instead, select points *conditional* on points already selected. Selecting points by *information* instead of by distance motivates conditional k-th nearest neighbors (Ck-NN).



Greedy mutual information maximization

Greedily select the next training point with highest mutual information with the target point. If I is the set of selected indices, select by:

$$I = I \cup \left\{ \underset{j \notin I}{\operatorname{argmax}} \frac{\Theta_{j,\text{Pr}\mid I}^2}{\Theta_{j,j\mid I}} \right\}$$

Efficient computation from Cholesky factor

A direct computation of the objective takes $\mathcal{O}(Nk^4)$ to select k points. This computational cost can be reduced to $\mathcal{O}(Nk^2)$ by storing a partial Cholesky factor, since each column is conditional on everything before it:

$$\mathrm{chol}(\Theta) = \begin{pmatrix} I & 0 \\ \Theta_{2,1}\Theta_{1,1}^{-1} & I \end{pmatrix} \begin{pmatrix} \mathrm{chol}(\Theta_{1,1}) & 0 \\ 0 & \mathrm{chol}(\Theta_{2,2} - \Theta_{2,1}\Theta_{1,1}^{-1}\Theta_{1,2}) \end{pmatrix}$$

Generalization to multiple prediction points

For multiple prediction points, the objective becomes to minimize the log determinant of the conditional covariance matrix of prediction points. By making use of the matrix determinant lemma, one can show that:

$$\operatorname{logdet}\left(\Theta_{\text{Pr},\text{Pr}\mid I\cup\{k\}}\right) - \operatorname{logdet}\left(\Theta_{\text{Pr},\text{Pr}\mid I}\right) = \log\left(\frac{\Theta_{kk\mid I,\text{Pr}}}{\Theta_{kk\mid I}}\right)$$

We can efficiently compute the objective by storing *two* Cholesky factors, yielding a complexity of $\mathcal{O}(Nk^2 + Nm^2 + m^3)$ for m prediction points.

Global approximation by KL-minimization

Approximate a Gaussian process by a sparse approximate Cholesky factor of its precision. Measure the resulting approximation accuracy by the KL divergence between the corresponding centered Gaussian processes:

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

Using the optimal unique minimizer L from closed form computation:

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

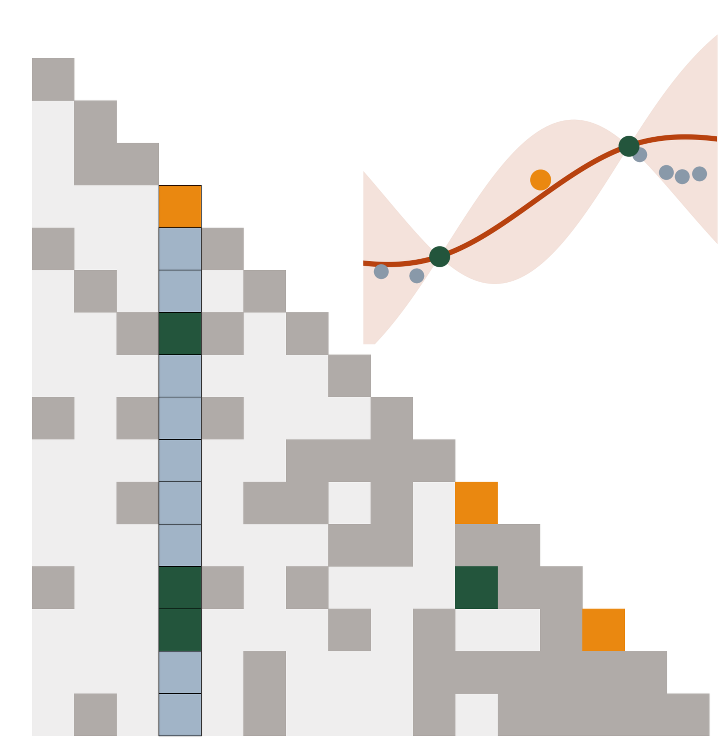
Objective becomes minimize variance of ith point, conditional on selected!

$$2\mathbb{D}_{\text{KL}}\left(\mathcal{N}(\mathbf{0}, \Theta) \,\middle\|\, \mathcal{N}(\mathbf{0}, (LL^\top)^{-1})\right) = \sum_{i=1}^{N} \left[\log\left(\Theta_{ii\mid s_i - \{i\}}\right)\right] - \operatorname{logdet}(\Theta)$$

Applying selection to Cholesky factorization

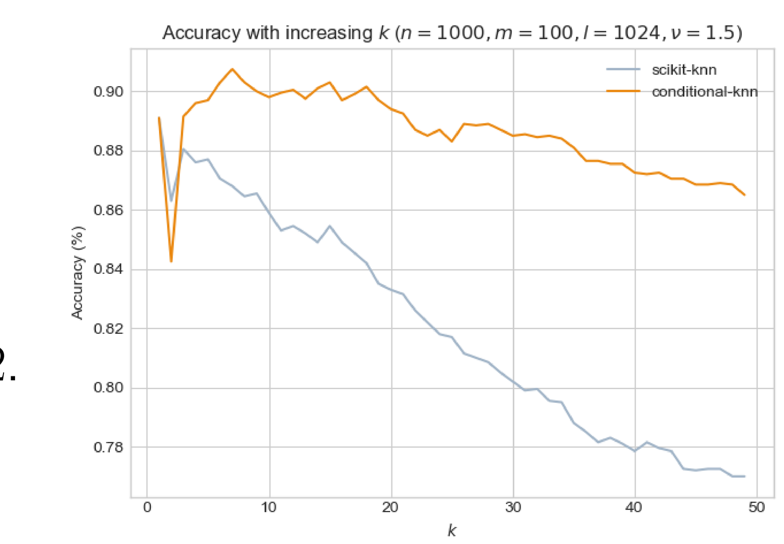
For a column in isolation, unknown point is the diagonal entry, below it are candidates, and selected entries to the sparsity pattern s_i.

However, for multiple aggregated columns (supernodes), a candidate can be added "partially" if it is between prediction points. By careful application of rank-one downdating, this structure can be preserved at no additional cost.



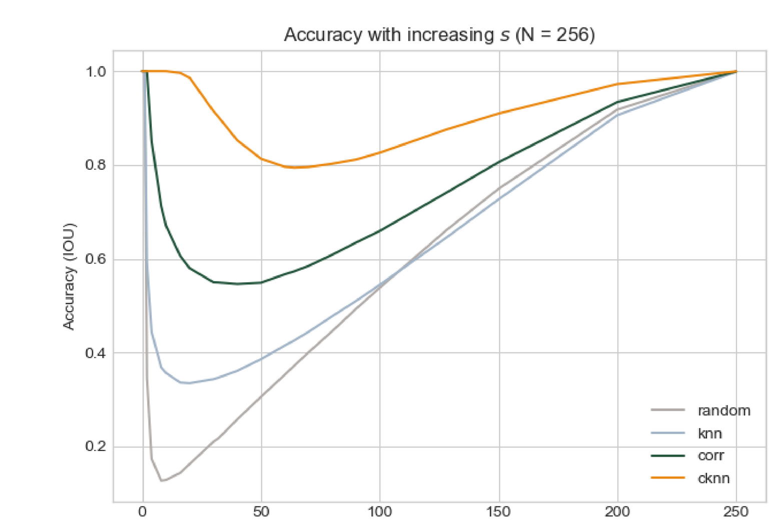
Drop-in replacement for k-NN on MNIST

We classify an image by taking the mode label in k selected images. Ck-NN gives better accuracies on the MNIST dataset for every $k > 2$.



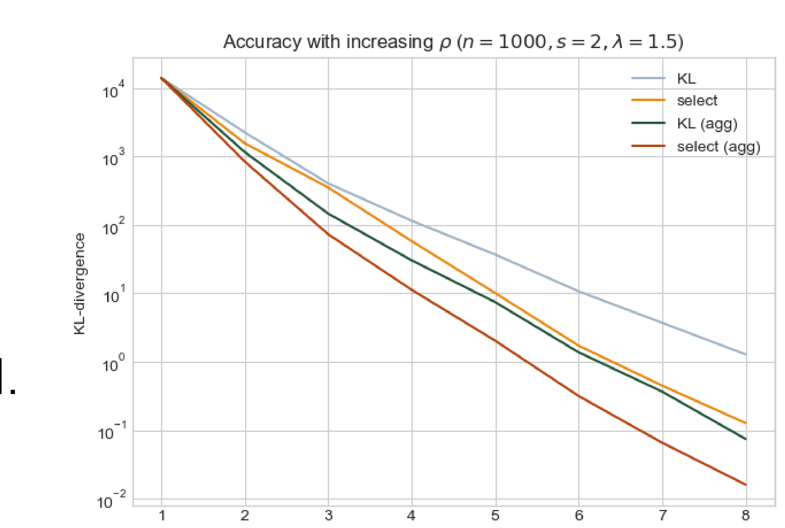
Recovery of sparse factors

We generate random sparse Cholesky factors by randomly selecting a prescribed number of nonzero entries per column. Ck-NN is able to recover the underlying sparse factor given its covariance with high accuracy.



Better KL divergence with sparser factors

Plugging the selection algorithm into Cholesky factorization leads to better KL divergence for the same number of nonzero entries as k-NN.



Preconditioning with conjugate gradient

Because the KL divergence strongly penalizes zero eigenvalues of the preconditioned matrix $L\Theta L^T$, the condition number is improved, resulting in less iterations with the conjugate gradient.