Scalable methods for large spatial data: Nearest Neighbor Gaussian processes

Abhi Datta

Department of Biostatistics, Bloomberg School of Public Health, Johns Hopkins University, Baltimore, Maryland

abhidatta.com @dattascience

Review of Low rank Gaussian Predictive Process

Pros

- Choose knots $S^* = \{s_1^*, s_2^*, \ldots, s_r^*\}$
- $\tilde{w}(s) = E(w(s) \mid w(S^*)) + \eta(s)$
- $var(\tilde{w}) = A_{n\times r} Var(\tilde{w}(S^*))_{r\times r} A' + D_{n\times n}$
- Matrix identities ensure we only need to invert the $r \times r$ matrix
- Computationally tractable – FLOPs count $O(nr^2)$
- Full Gaussian process (with a low-rank plus diagonal covariance function)
- Allows for coherent spatial interpolation at arbitrary resolution
- Can be used as prior for spatial random effects in any hierarchical setup for spatial data

Review of Low rank Gaussian Predictive Process

Cons

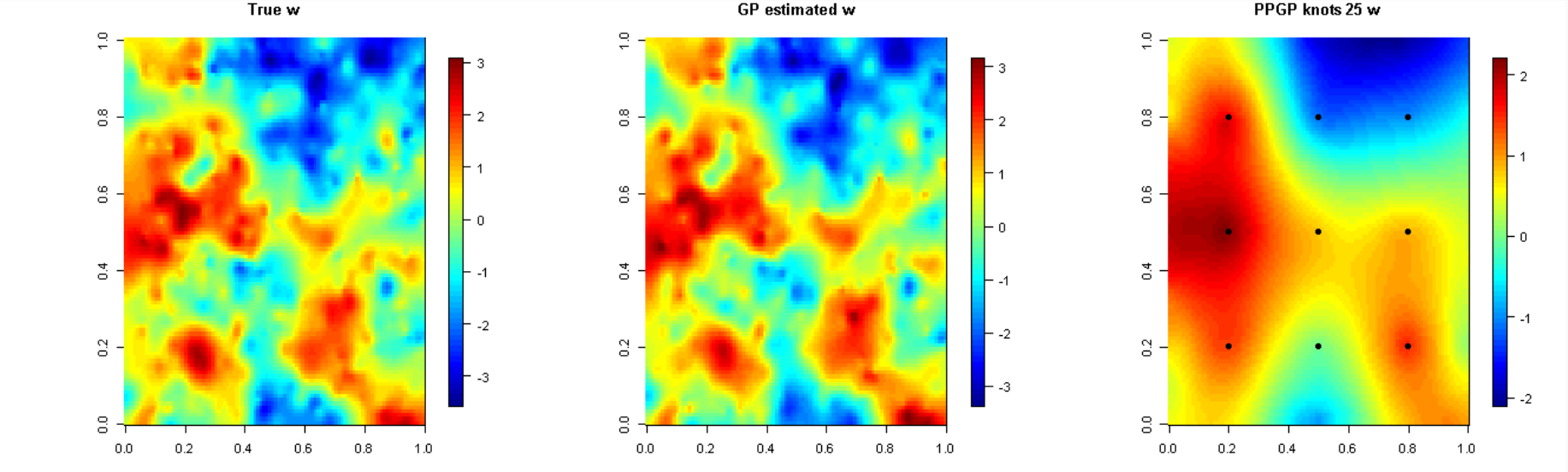


Figure: Comparing full GP vs low-rank GP

- Low rank models like the Predictive Process (PP) often tends to oversmooth
- Increasing the number of knots can fix this but will lead to heavy computation

Sparse matrices

- Idea: Use a sparse matrix instead of a low rank matrix to approximate the dense full GP covariance matrix
- Goals:
  - Scalability: Both in terms of storage and computing inverse and determinants
  - Closely approximate full GP inference
  - Proper Gaussian process model like the Predictive Process

Cholesky factors

- Write a joint density $p(w) = p(w_1, w_2, \ldots, w_n)$ as:

$$p(w_1)p(w_2 \mid w_1)p(w_3 \mid w_1, w_2) \cdots p(w_n \mid w_1, w_2, \ldots, w_{n-1})$$

- For Gaussian distribution $w \sim N(0, C)$ this $\Rightarrow$

$$\begin{aligned}
w_1 &= 0 + \eta_1; \\
w_2 &= a_{21} w_1 + \eta_2; \\
\ldots &\quad \ldots \quad \ldots \\
w_n &= a_{n1} w_1 + a_{n2} w_2 + \cdots + a_{n,n-1} w_{n-1} + \eta_n;
\end{aligned}$$

Cholesky factors

- Write a joint density $p(w) = p(w_1, w_2, \ldots, w_n)$ as:

$$p(w_1)p(w_2 \mid w_1)p(w_3 \mid w_1, w_2) \cdots p(w_n \mid w_1, w_2, \ldots, w_{n-1})$$

- For Gaussian distribution $w \sim N(0, C)$ this $\Rightarrow$

$$\begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_n \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & \ldots & 0 & 0 \\ a_{21} & 0 & 0 & \ldots & 0 & 0 \\ a_{31} & a_{32} & 0 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \ldots & a_{n,n-1} & 0 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_n \end{bmatrix} + \begin{bmatrix} \eta_1 \\ \eta_2 \\ \eta_3 \\ \vdots \\ \eta_n \end{bmatrix}$$

$$\Longrightarrow w = Aw + \eta; \quad \eta \sim N(0, D), \text{ where } D = diag(d_1, d_2, \ldots, d_n).$$

Cholesky factors

- Write a joint density $p(w) = p(w_1, w_2, \ldots, w_n)$ as:

$$p(w_1)p(w_2 \mid w_1)p(w_3 \mid w_1, w_2) \cdots p(w_n \mid w_1, w_2, \ldots, w_{n-1})$$

- For Gaussian distribution $w \sim N(0, C)$ this $\Rightarrow$

$$\begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_n \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & \ldots & 0 & 0 \\ a_{21} & 0 & 0 & \ldots & 0 & 0 \\ a_{31} & a_{32} & 0 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \ldots & a_{n,n-1} & 0 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_n \end{bmatrix} + \begin{bmatrix} \eta_1 \\ \eta_2 \\ \eta_3 \\ \vdots \\ \eta_n \end{bmatrix}$$

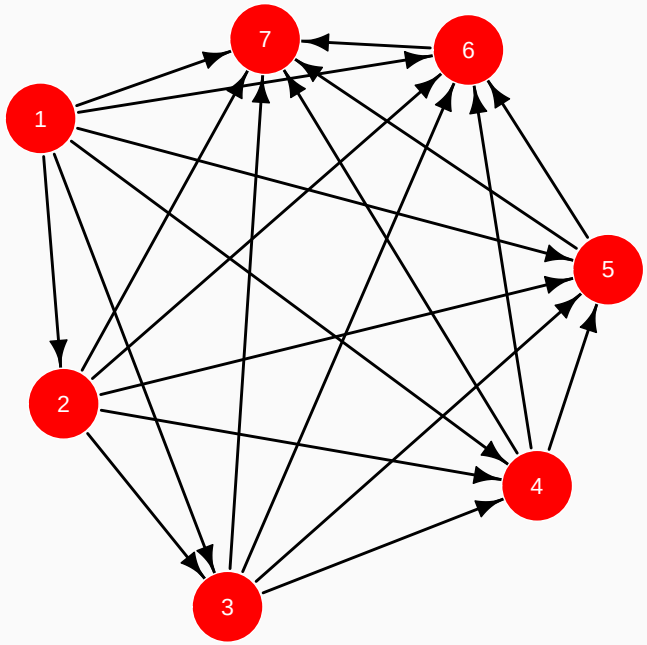
$\Longrightarrow w = Aw + \eta; \quad \eta \sim N(0, D)$, where $D = diag(d_1, d_2, \ldots, d_n)$.

- Cholesky factorization: $C = (I - A)^{-1} D (I - A)^{-\top}$ or $C^{-1} = (I - A)' D^{-1} (I - A)$

Cholesky factors

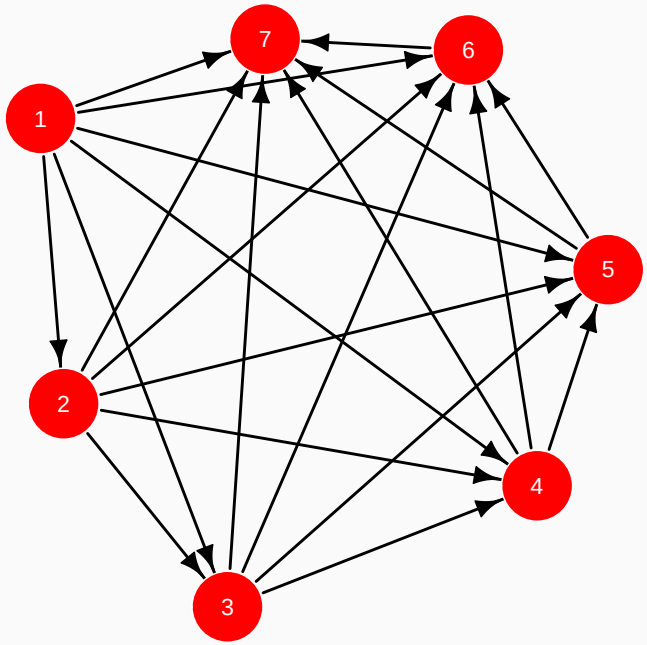
- $w_{<i} = (w_1, w_2, \ldots, w_{i-1})'$
- $c_i = Cov(w_i, w_{<i})$, $C_i = Var(w_{<i})$
- i^{th} row of A and $d_i = Var(\eta_i)$ are obtained from $p(w_i \mid w_{<i})$ as follows:
  - Solve for a_{ij}'s from $\sum_{j=1}^{i-1} a_{ij} w_j = E(w_i \mid w_{<i}) = c_i' C_i^{-1} w_{<i}$
  - $d_i = Var(w_i \mid w_{<i}) = \sigma^2 - c_i' C_i^{-1} c_i$
- For large i, inverting C_i becomes slow
- The Cholesky factor approach for the full GP covariance matrix C does not offer any computational benefits

Cholesky Factors and Directed Acyclic Graphs (DAGs)



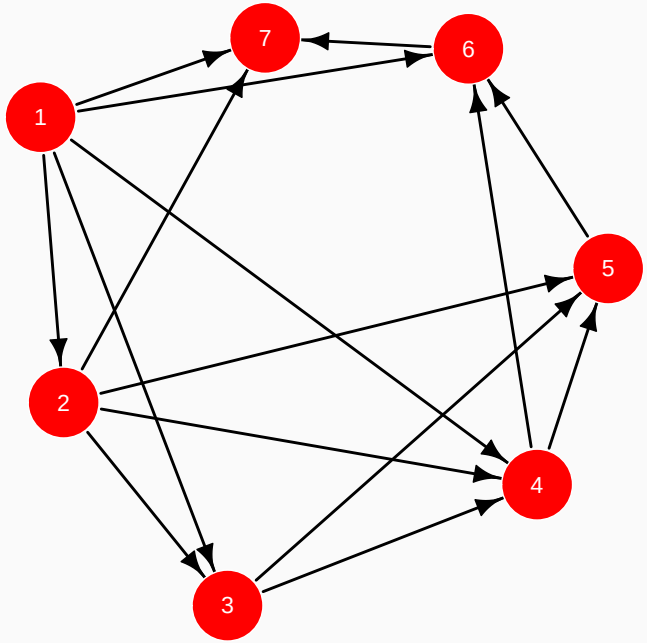
- Number of non-zero entries (sparsity) of A equals number of arrows in the graph
- In particular: Sparsity of the i^{th} row of A is same as the number of arrows towards i in the DAG

Introducing sparsity via graphical models



$$p(y_1)p(y_2 \mid y_1)p(y_3 \mid y_1, y_2)p(y_4 \mid y_1, y_2, y_3)$$
$$\times\, p(y_5 \mid y_1, y_2, y_3, y_4)p(y_6 \mid y_1, y_2, \ldots, y_5)p(y_7 \mid y_1, y_2, \ldots, y_6)\,.$$

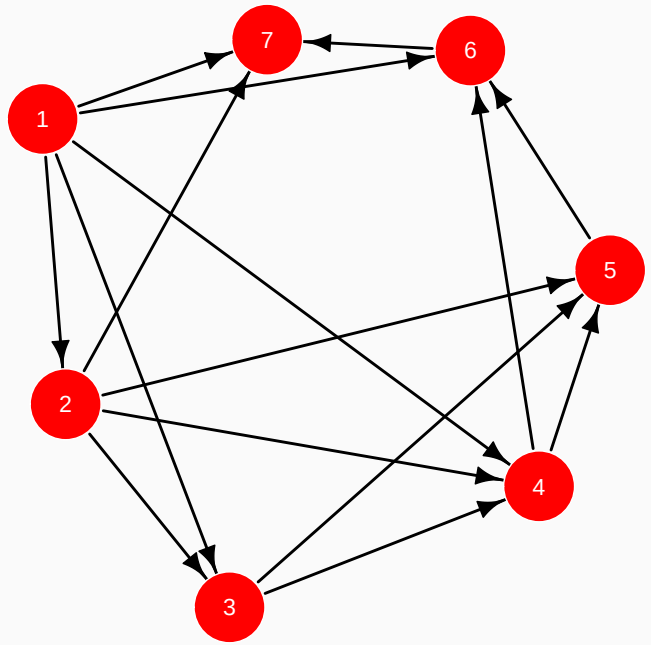
Introducing sparsity via graphical models



$$p(y_1)p(y_2 \mid y_1)p(y_3 \mid y_1, y_2)p(y_4 \mid y_1, y_2, y_3)$$

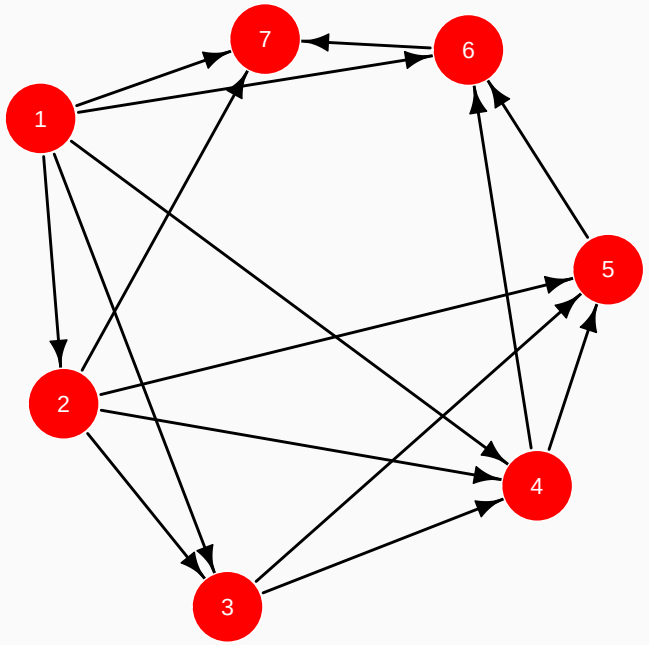
$$p(y_5 \mid \cancel{y_1}, y_2, y_3, y_4)p(y_6 \mid y_1, \cancel{y_2}, \cancel{y_3}, y_4, y_5)p(y_7 \mid y_1, y_2, \cancel{y_3}, \cancel{y_4}, \cancel{y_5}, y_6)$$

Introducing sparsity via graphical models



- Create a sparse DAG by keeping at most m arrows pointing to each node
- Set $a_{ij} = 0$ for all i, j which has no arrow between them
- Fixing $a_{ij} = 0$ introduces conditional independence and w_j drops out from the conditional set in $p(w_i \mid \{w_k : l < i\})$

Introducing sparsity via graphical models



- $N(i)$ denote *neighbor set* of i, i.e., the set of nodes from which there are arrows to i
- $a_{ij} = 0$ for $j \notin N(i)$ and nonzero a_{ij}'s obtained by solving:

$$\mathrm{E}[w_i \mid w_{N(i)}] = \sum_{j \in N(i)} a_{ij} w_j = c_{i,N(i)} C_{N(i)}^{-1} w_{N(i)}$$

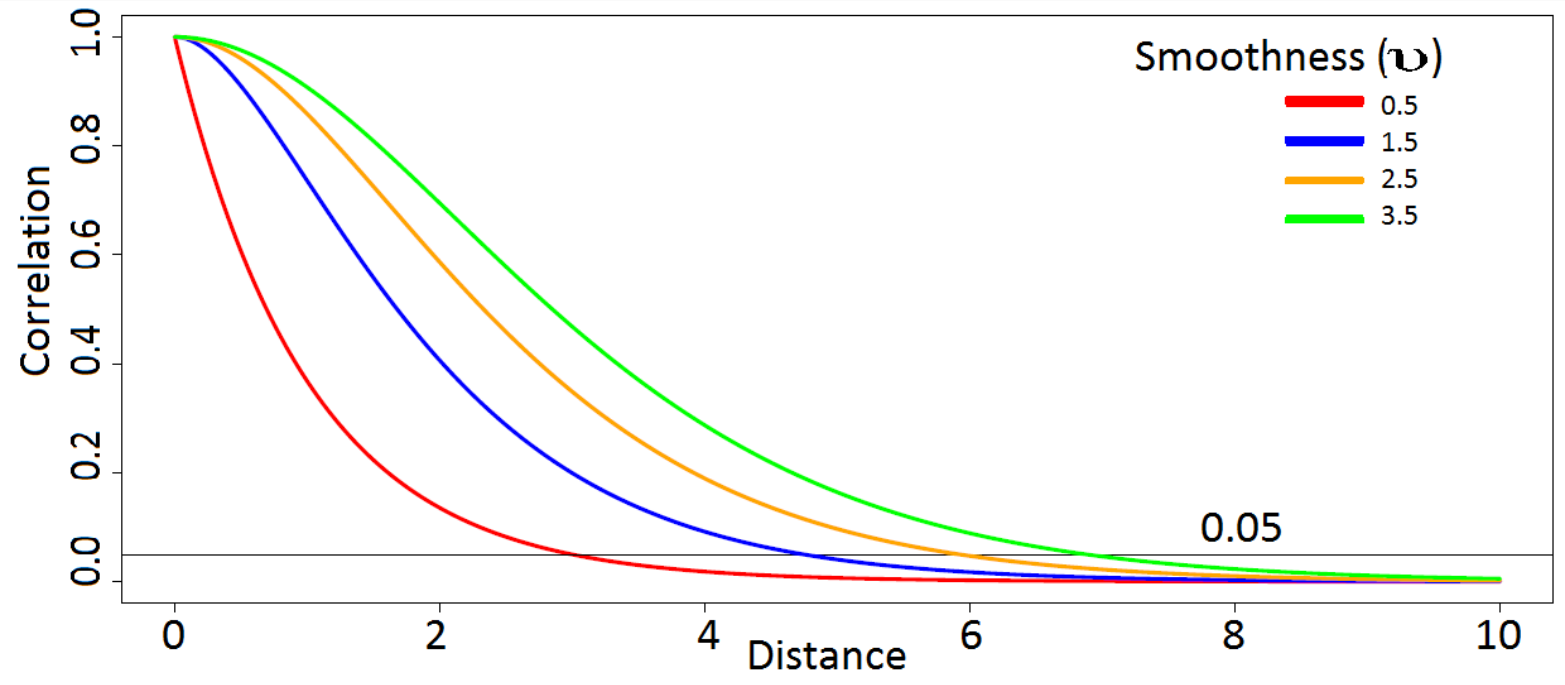
$$\mathrm{Var}[w_i \mid w_{N(i)}] = d_i = \sigma_i^2 - c_{i,N(i)} C_{N(i)}^{-1} c_{N(i),i}$$

- The above matrix is only $m \times m$

Choosing neighbor sets

Matern Covariance Function:

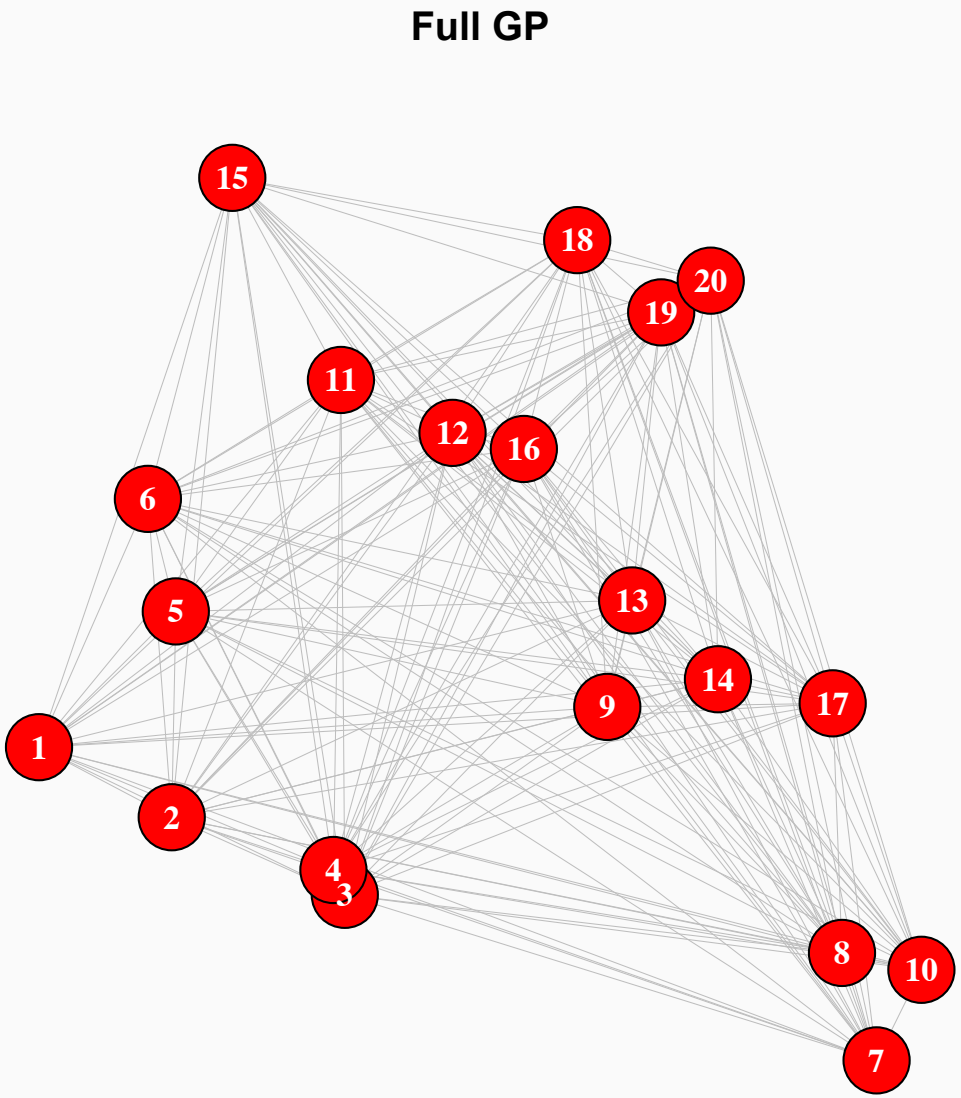
$$C(s_i, s_j) = \frac{1}{2^{\nu-1}\Gamma(\nu)}(||s_i - s_j||\phi)^{\nu}\mathscr{K}_{\nu}(||s_i - s_j||\phi);\ \phi > 0, \nu > 0,$$



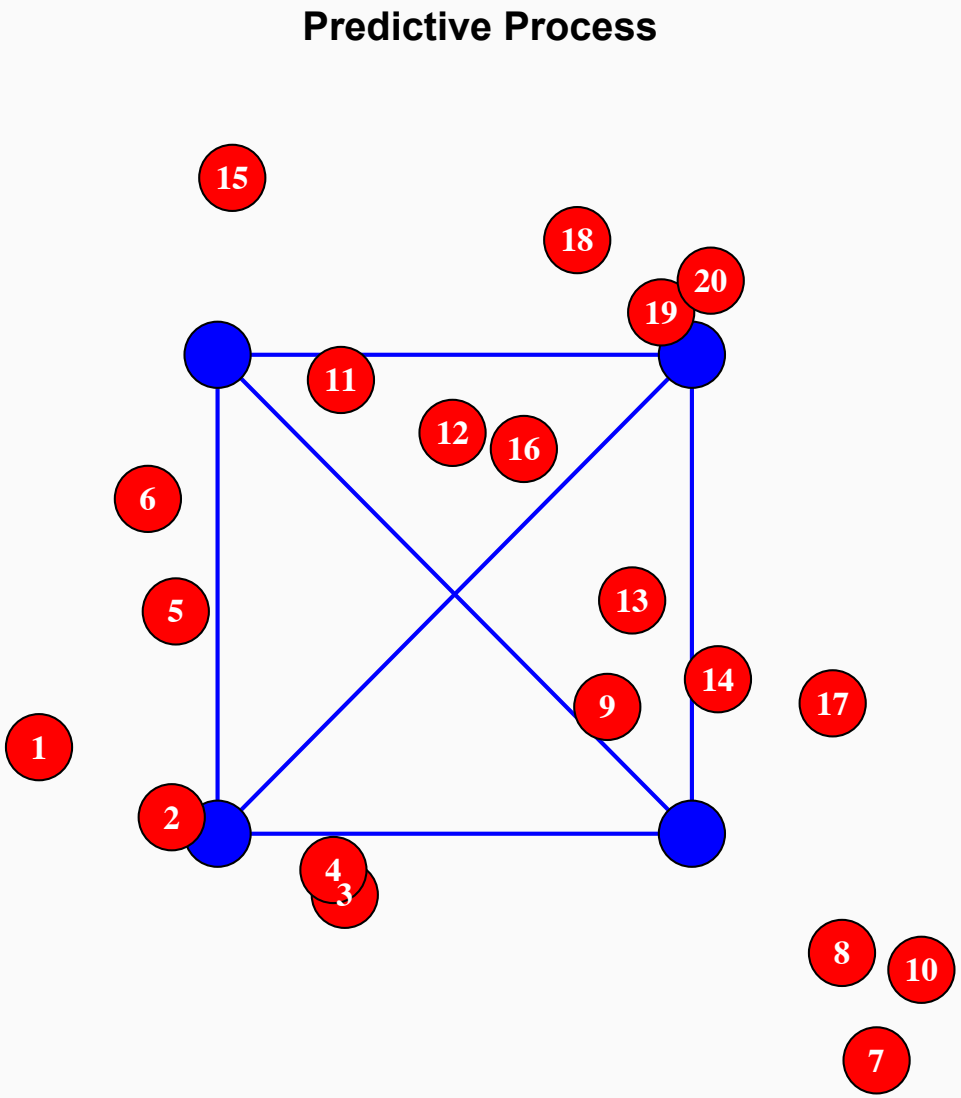
Choosing neighbor sets

- Spatial covariance functions decay with distance
- Vecchia (1988): $N(s_i) = m-$nearest neighbors of s_i in $s_1, s_2, \ldots, s_{i-1}$
    - Nearest points have highest correlations
    - Theory: "Screening effect" – Stein, 2002
- We use Vecchia's choice of m-nearest neighbor
- Other choices proposed in Stein et al. (2004); Gramacy and Apley (2015); Guinness (2016) can also be used

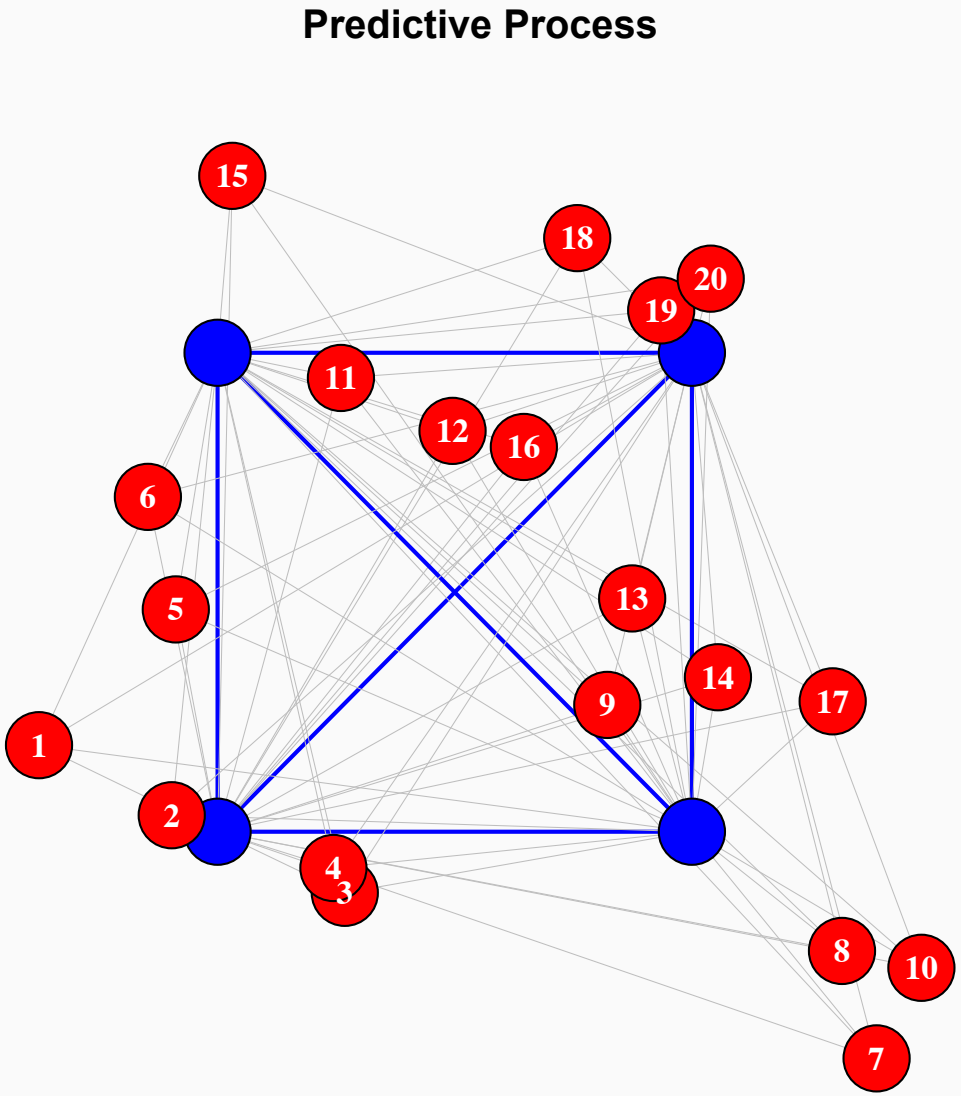
Graohical models: Full GP vs low rank vs NNGP



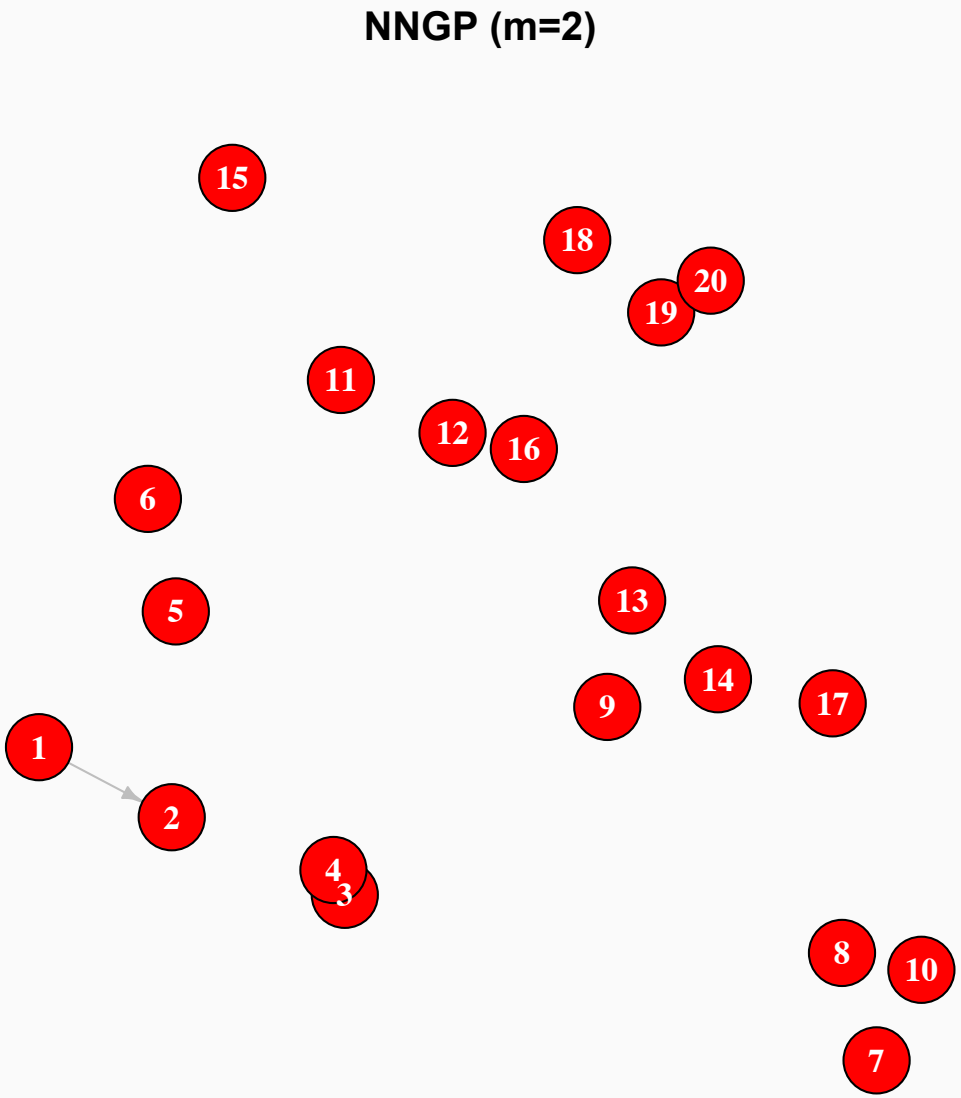
Graohical models: Full GP vs low rank vs NNGP



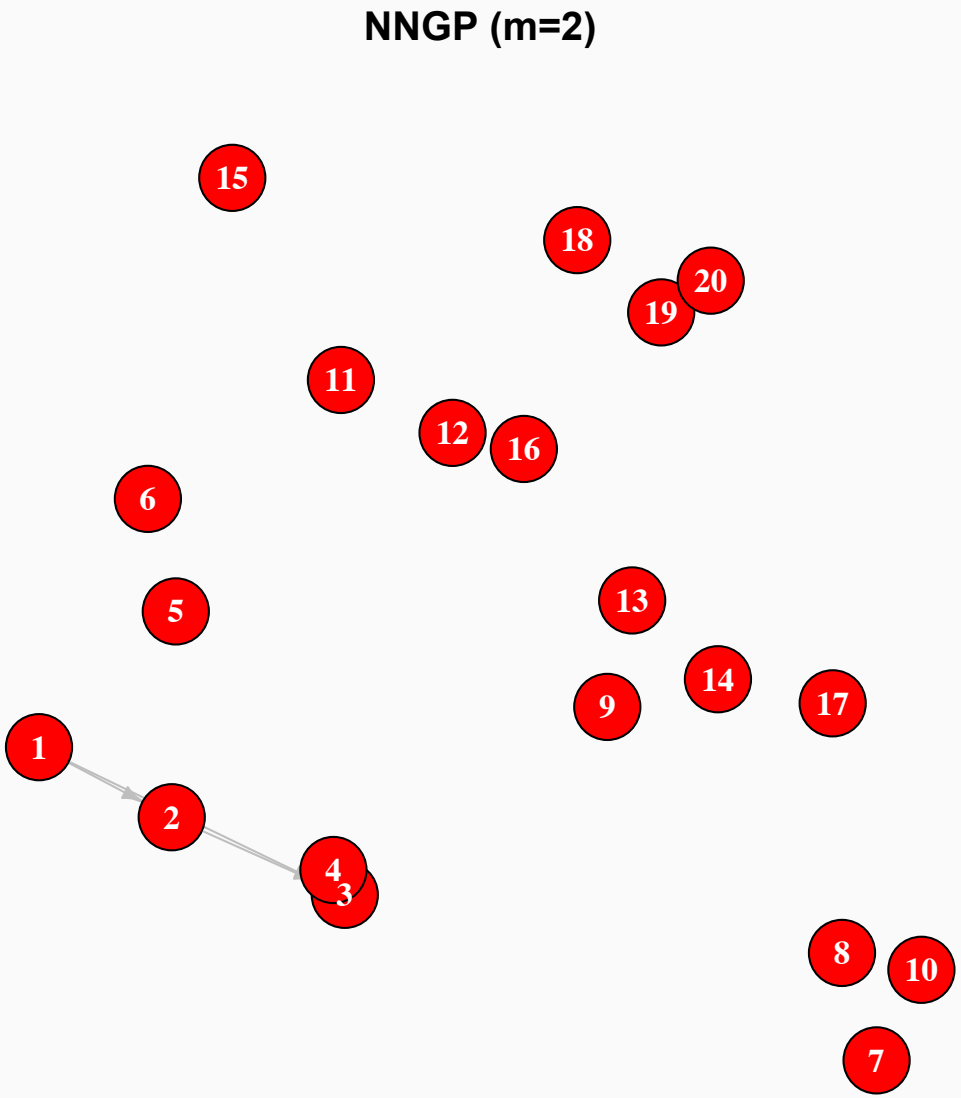
Graohical models: Full GP vs low rank vs NNGP



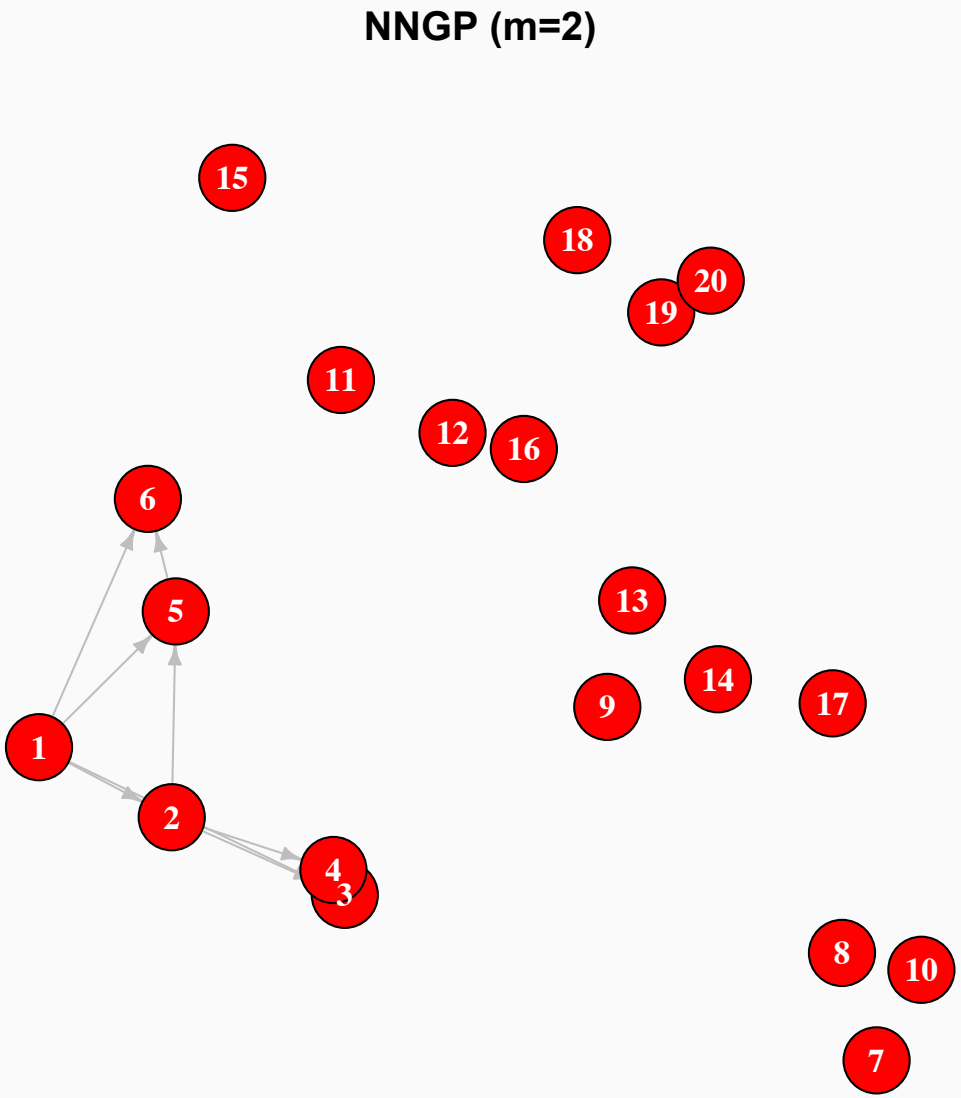
Graohical models: Full GP vs low rank vs NNGP



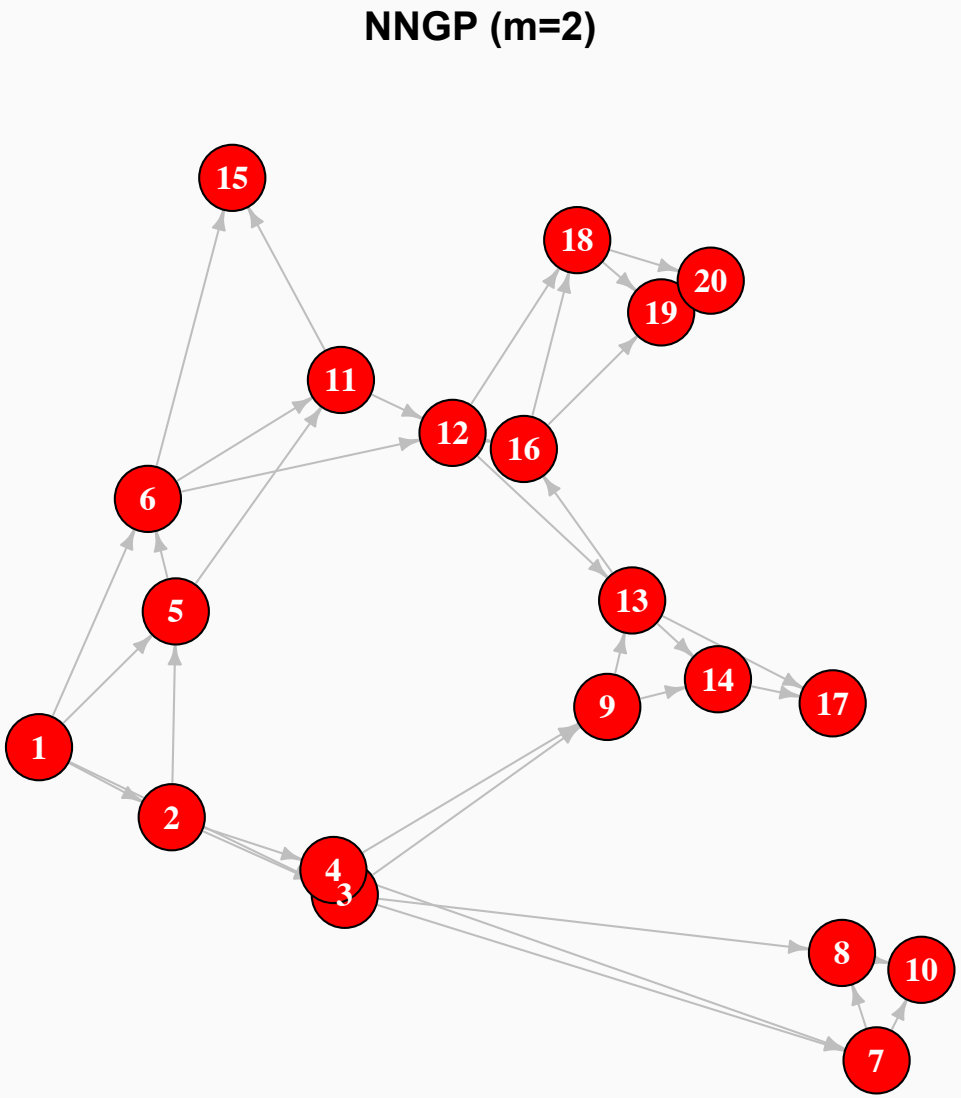
Graohical models: Full GP vs low rank vs NNGP



Graohical models: Full GP vs low rank vs NNGP

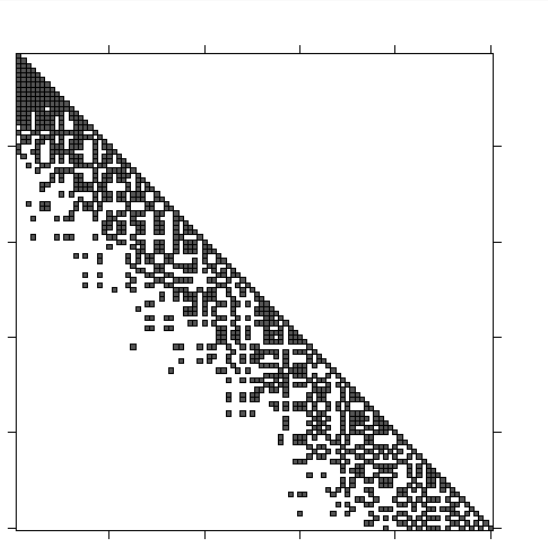


Graohical models: Full GP vs low rank vs NNGP

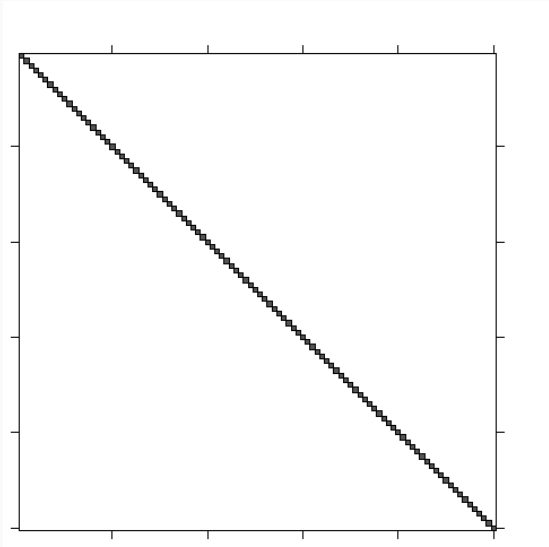


Sparse precision matrices

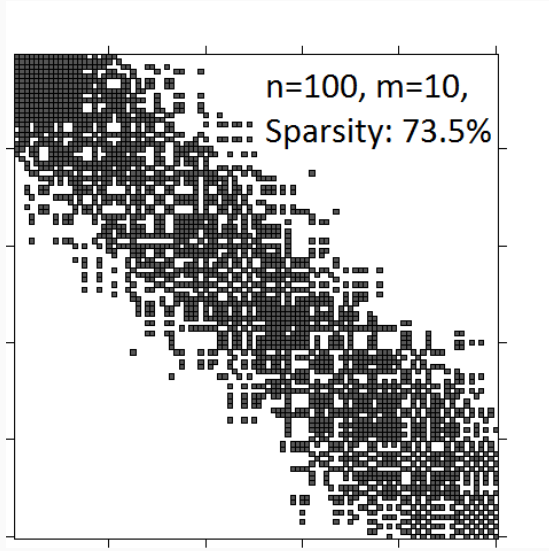
- The neighbor sets and the covariance function $C(\cdot, \cdot)$ define a sparse Cholesky factor $D^{-1/2}(I - A)$ of the spatial precision (inverse-covariance) matrix
- $N(w \mid 0, C) \approx N(w \mid 0, \tilde{C})$; $\tilde{C}^{-1} = (I - A)^\top D^{-1}(I - A)$



$D^{-1/2}(I - A)$



D

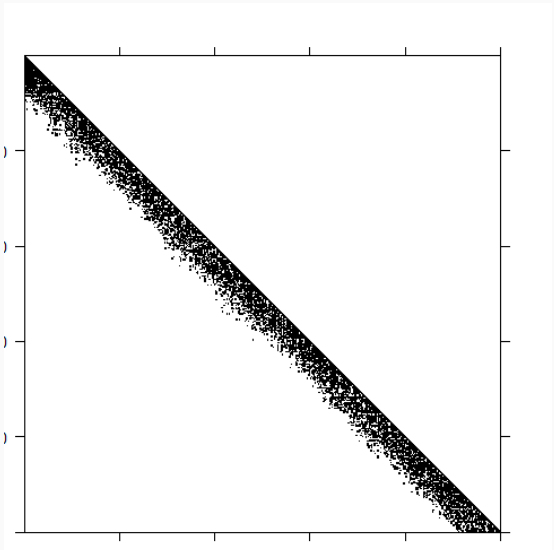


$\tilde{C}^{-1}$

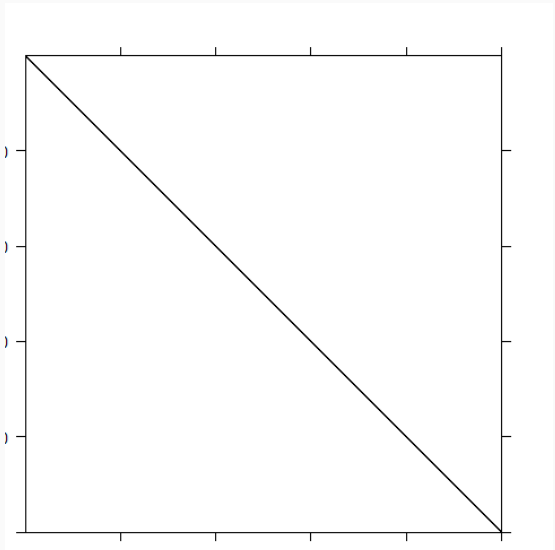
- $\det(\tilde{C}) = \prod_{i=1}^{n} D_i$,
- $\tilde{C}^{-1}$ is sparse with $O(nm^2)$ entries

Sparse precision matrices

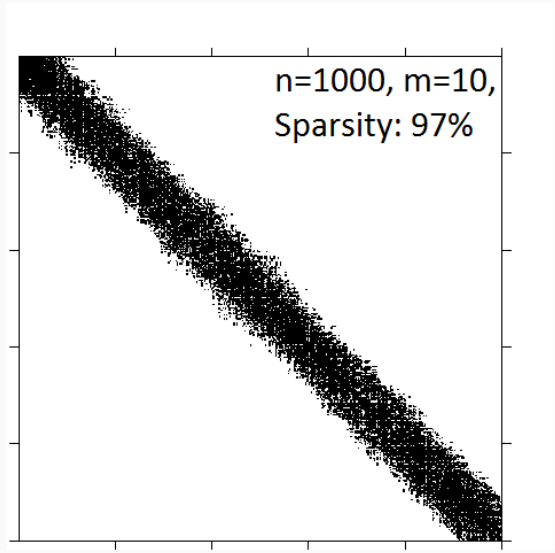
- The neighbor sets and the covariance function $C(\cdot, \cdot)$ define a sparse Cholesky factor $D^{-1/2}(I - A)$ of the spatial precision (inverse-covariance) matrix
- $N(w \mid 0, C) \approx N(w \mid 0, \tilde{C})$; $\tilde{C}^{-1} = (I - A)^\top D^{-1}(I - A)$



$D^{-1/2}(I - A)$



D



$\tilde{C}^{-1}$

- $\det(\tilde{C}) = \prod_{i=1}^{n} D_i$,
- $\tilde{C}^{-1}$ is sparse with $O(nm^2)$ entries

Extension to a Process

- We have defined $w \sim N(0, \tilde{C})$ over the set of data locations $S = \{s_1, s_2, \ldots, s_n\}$
- For $s \notin S$, define $N(s)$ as set of m-nearest neighbors of s in S
- Define $w(s) = \sum_{i:s_i \in N(s)} a_i(s) w(s_i) + \eta(s)$ where $\eta(s) \overset{ind}{\sim} N(0, d(s))$
  - $a_i(s)$ and $d(s)$ are once again obtained by solving $m \times m$ system respectively as the m-nearest neighbor kriging mean weights and kriging variance
- Well-defined GP over entire domain
  - Nearest Neighbor GP (NNGP) – Datta et al. (2016)[1]

[1] https://www.tandfonline.com/doi/abs/10.1080/01621459.2015.1044091

Hierarchical spatial regression with NNGP

Spatial linear model

$$y(\mathbf{s}) = x(\mathbf{s})'\beta + w(\mathbf{s}) + \epsilon(\mathbf{s})$$

- $w(s)$ modeled as *NNGP* derived from a $GP(0, C(\cdot, \cdot, \,|\, \sigma^2, \phi))$
- $\epsilon(s) \overset{\text{iid}}{\sim} N(0, \tau^2)$ contributes to the nugget
- Priors for the parameters β, σ^2, τ^2 and ϕ
- Only difference from a full GP model is the NNGP prior $w(s)$

Hierarchical spatial regression with NNGP

Full Bayesian Model

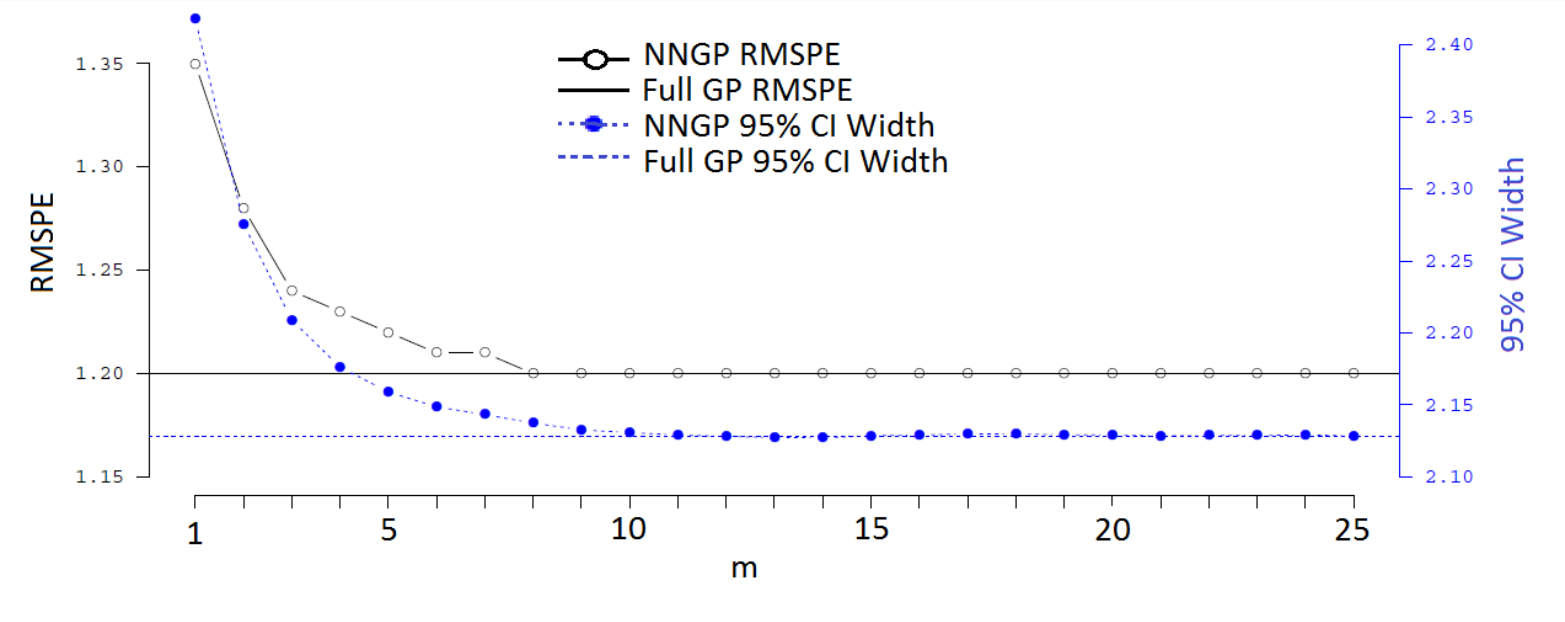
$$N(y \mid X\beta + w, \tau^2 I) \times N(w \mid 0, \tilde{C}(\sigma^2, \phi)) \times N(\beta \mid \mu_\beta, V_\beta)$$
$$\times IG(\tau^2 \mid a_\tau, b_\tau) \times IG(\sigma^2 \mid a_\sigma, b_\sigma) \times Unif(\phi \mid a_\phi, b_\phi)$$

Gibbs sampler:

- Conjugate full conditionals for β, τ^2, σ^2 and $w(s_i)$'s
- Metropolis step for updating ϕ
- Posterior predictive distribution at any location using composition sampling:

$$\int N(y(s) \mid x(s)'\beta + w(s), \tau^2 I) \times N(w(s) \mid a(s)'w_R, d(s)) \times$$
$$p(w, \beta, \tau^2, \sigma^2, \phi \mid y)\, d(w, \beta, \tau^2, \sigma^2, \phi)$$

Choosing m



- Run NNGP in parallel for few values of m
- Choose m based on model evaluation metrics
- Our results suggested that typically $m \approx 20$ yielded excellent approximations to the full GP

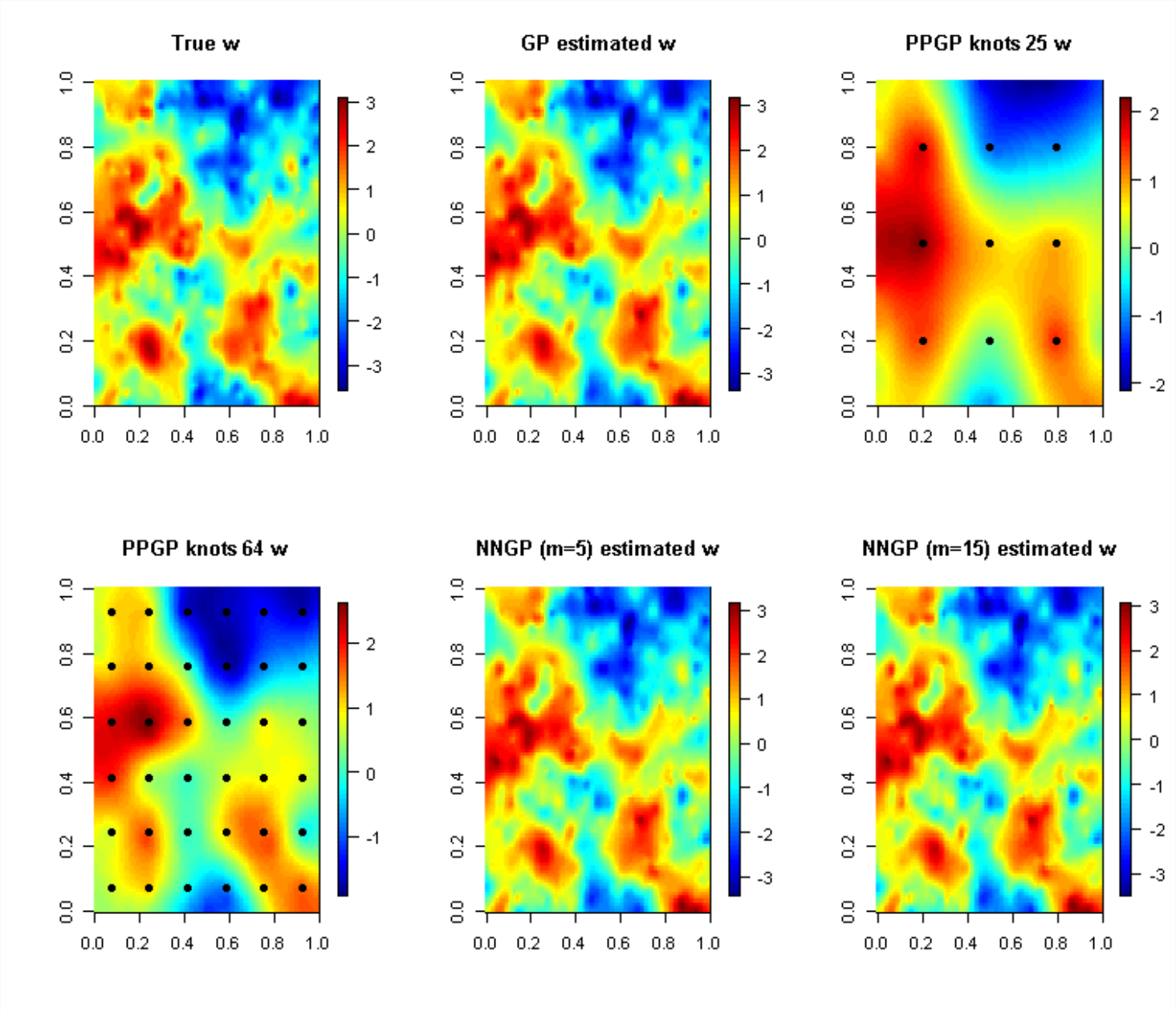
Storage and computation

- Storage:
  - Never needs to store $n \times n$ distance matrix
  - Stores smaller $m \times m$ matrices
  - Total storage requirements $O(nm^2)$
- Computation:
  - Only involves inverting small $m \times m$ matrices
  - Total flop count per iteration of Gibbs sampler is $O(nm^3)$
- Since $m \ll n$, NNGP offers great scalability for large datasets

spNNGP package in R

- Implements the MCMC for spatial regression model using NNGP
- Full posterior distributions of all parameters available (similar to spBayes)
- Suitable for parallel computing
- Implements NNGP variants like the response model and the MCMC-free conjugate model
- Very suitable for analyzing very large spatial datasets (upto millions of locations)

Predicted surfaces of w



Reducing parameter dimensionality

- The Gibbs sampler algorithm for the NNGP updates $w(s_1), w(s_2), \ldots, w(s_n)$ sequentially
- Dimension of the MCMC for this sequential algorithm is $O(n)$
- If the number of data locations n is very large, this high-dimensional MCMC can converge slowly
- Although each iteration for the NNGP model will be very fast, many more MCMC iterations may be required
- Solution: Back to the marginalized model?

Collapsed NNGP

- Same model:

$$y(s) = x(s)'\beta + w(s) + \epsilon(s)$$
$$w(s) \sim NNGP(0, C(\cdot, \cdot \mid \theta))$$
$$\epsilon(s) \overset{\text{iid}}{\sim} N(0, \tau^2)$$

- Vector form $y \sim N(X\beta + w, \tau^2 I); w \sim N(0, \tilde{C}(\theta))$
- Collapsed model: Marginalizing out w, we have $y \sim N(X\beta, \tau^2 I + \tilde{C}(\theta))$

Collapsed NNGP

Model

$y \sim N(X\beta, \tau^2 I + \tilde{C}(\theta))$

- Only involves few parameters β, τ^2 and $\theta = (\sigma^2, \phi)'$
- Drastically reduces the MCMC dimensionality
- Gibbs sampler updates are based on sparse linear systems using $\tilde{C}^{-1}$
- Improved MCMC convergence
- Can recover posterior distribution of $w \mid y$
- Complexity of the algorithm depends on the design of the data locations and is not guaranteed to be $O(n)$

Response NNGP

- $w(s) \sim GP(0, C(\cdot, \cdot \mid \theta)) \Rightarrow y(s) \sim GP(x(s)'\beta, \Sigma(\cdot, \cdot \mid \tau^2, \theta))$
- $\Sigma(s_i, s_j) = C(s_i, s_j \mid \theta) + \tau^2 \, \delta(s_i = s_j)$ (δ is Kronecker delta)
- We can directly derive the NNGP covariance function corresponding to $\Sigma(\cdot, \cdot)$
- $\tilde{\Sigma}$ is the NNGP covariance matrix for the n locations
- Response model: $y \sim N(X\beta, \tilde{\Sigma})$
- Storage and computations are guaranteed to be $O(n)$
- Low dimensional MCMC $\Rightarrow$ Improved convergence
- Cannot coherently recover $w \mid y$

MCMC convergence

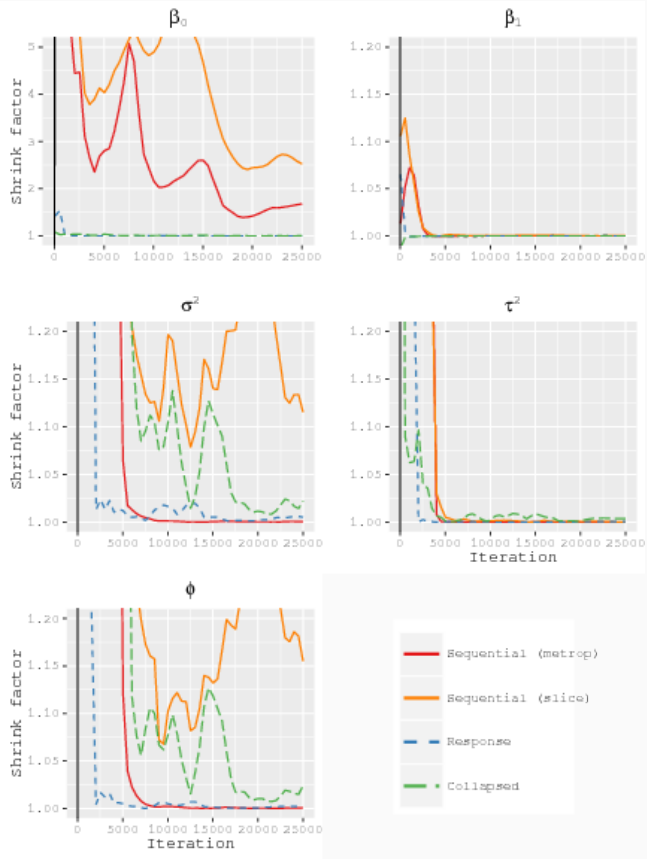


Figure: MCMC convergence diagnostics using Gelman-Rubin shrink factor for different NNGP models [2] for a simulated dataset

[2] Finley et al. (2019): https://www.tandfonline.com/doi/abs/10.1080/10618600.2018.1537924?journalCode=ucgs20

Conjugate NNGP

- Original full GP model: $y(s) \overset{ind}{\sim} N(x(s)'\beta + w(s), \tau^2)$
- $w(s) \sim GP$ with a stationary covariance function $C(\cdot, \cdot \mid \sigma^2, \phi)$
- $Cov(w) = \sigma^2 R(\phi)$
- Full GP model: $y \sim N(X\beta, \Sigma)$ where $\Sigma = \sigma^2 M$
- $M = R(\phi) + \alpha I$
- $\alpha = \tau^2/\sigma^2$ is the ratio of the noise to signal variance
- Response NNGP model: $y \sim N(X\beta, \tilde{\Sigma})$
- $\tilde{\Sigma} = \sigma^2 \tilde{M}$ where $\tilde{M}$ is the NNGP approximation for M

Conjugate NNGP

- $y \sim N(X\beta, \sigma^2 \tilde{M})$
- If ϕ and α are known, M, and hence $\tilde{M}$, are known matrices
- The model becomes a standard Bayesian linear model
- Assume a *Normal Inverse Gamma (NIG)* prior for $(\beta, \sigma^2)'$
- $(\beta, \sigma^2)' \sim NIG(\mu_\beta, V_\beta, a_\sigma, b_\sigma)$, i.e., $\beta \,|\, \sigma^2 \sim N(\mu_\beta, \sigma^2 V_\beta)$ and $\sigma^2 \sim IG(a_\sigma, b_\sigma)$

Conjugate NNGP

- $y \sim N(X\beta, \sigma^2\tilde{M})$, $\tilde{M}$ is known

Joint likelihood:

$$N(y \mid X\beta, \sigma^2\tilde{M}) \times N(\beta \mid \mu_\beta, \sigma^2 V_\beta) \times IG(\sigma^2 \mid a_\sigma, b_\sigma)$$

Conjugate NNGP

- $y \sim N(X\beta, \sigma^2\tilde{M})$, $\tilde{M}$ is known

Joint likelihood:

$$N(y \mid X\beta, \sigma^2\tilde{M}) \times N(\beta \mid \mu_\beta, \sigma^2 V_\beta) \times IG(\sigma^2 \mid a_\sigma, b_\sigma)$$

- Conjugate posterior distribution
  $(\beta, \sigma^2) \mid y \sim NIG(\mu^*_\beta, V^*_\beta, a^*_\sigma, b^*_\sigma)$
- μ^*_β, V^*_β, a^*_σ and b^*_σ can be calculated in $O(n)$ time
- Exact posterior distributions, means, and variances of β and σ^2 are available in closed form
- $y(s) \mid y \sim t_{2a^*_\sigma}(m(s), \frac{b^*_\sigma}{a^*_\sigma} v(s))$, i.e., follows a t-distribution
- Exact posterior predictive distributions of $y(s) \mid y$ for any s
- No MCMC required for parameter estimation or prediction

Choosing α and ϕ

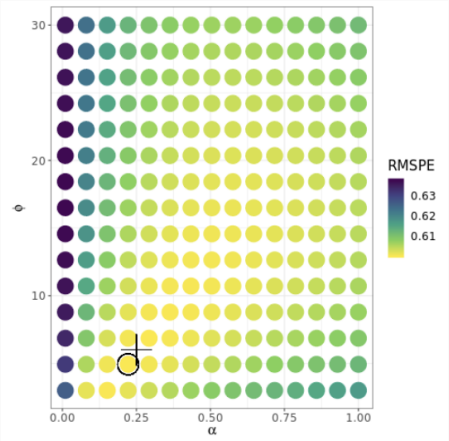


Figure: Simulation experiment in Finley et al, 2020[3]: True value ($+$) of (α, ϕ) and estimated value ($\circ$) using 5-fold cross validation

- ϕ and α are chosen using K-fold cross validation over a grid of possible values
- Unlike MCMC, cross-validation can be completely parallelized
- Resolution of the grid for ϕ and α can be decided based on computing resources available
- In practice, a reasonably coarse grid often suffices

[3] https://arxiv.org/pdf/2001.09111.pdf

Scalability

- Computation and storage requirements are $O(n)$
- One evaluation time similar to the response NNGP model
- Unlike response NNGP, does not involve any serial MCMC iterations
- For K fold cross validation and G combinations of ϕ and α, total number of evaluations is KG
- Embarassingly parallel: Each of the KG evaluations can proceed in parallel

Statutory figure on embarrassingly parallel computing

Comparison of run times

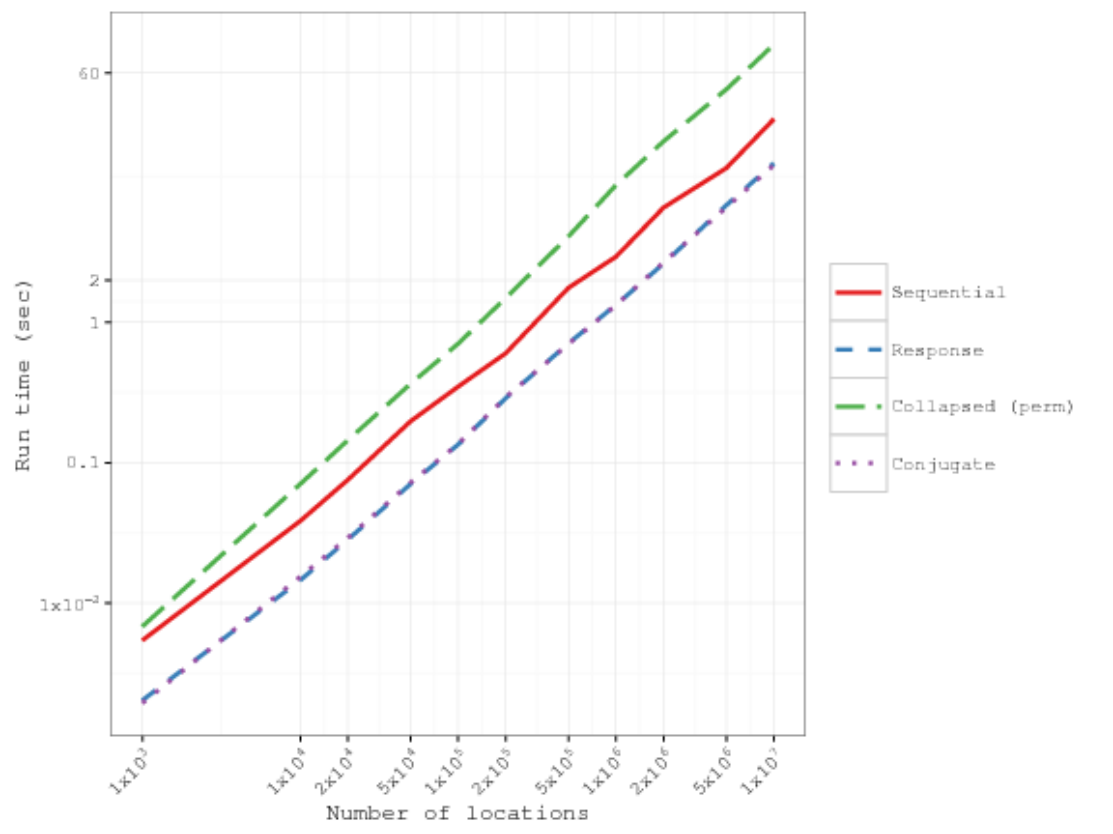


Figure: Run times of different NNGP models with increasing sample size for a simulated dataset[4]

[4] Finley et al. (2019): https://www.tandfonline.com/doi/abs/10.1080/10618600.2018.1537924?journalCode=ucgs20

Comparison of Bayesian NNGP models

	Sequential	Collapsed	Response	Conjugate
$O(n)$ time	Yes	No	Yes	Yes
Recovery of $w \mid y$	Yes	Yes	No	No
MCMC dimensionality	High	Low	Low	MCMC-free
Fully Bayesian inference	Yes	Yes	Yes	No
Embarassingly parallel	No	No	No	Yes

Comparing predictions of different NNGP models

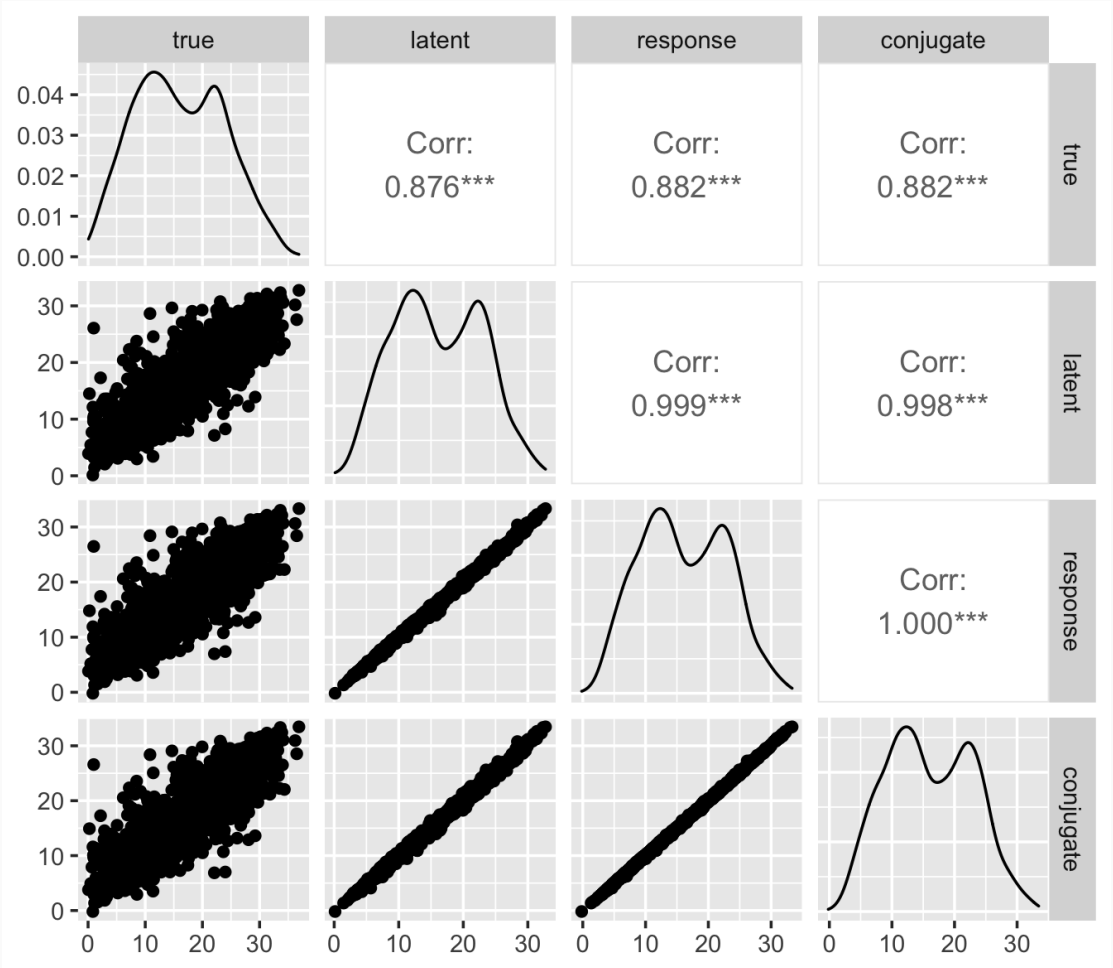


Figure: Comparing hold-out predictions from of forest canopy height different NNGP models using the Bonanza Creek Experimental Forest dataset of spNNGP package

The BRISC package[6]

- Uses maximum-likelihood based estimation from the response NNGP model
- `BRISC_estimation` for parameter estimation
- `BRISC_prediction` for spatial predictions using NNGP
- `BRISC_bootstrap` – NNGP-based Bootstrap for Rapid Inference on Spatial Covariances (Saha and Datta, 2018)[5] to obtain parameter confidence intervals
- New feature: `BRISC_simulation` for fast approximate simulation of large scale Gaussian Process realizations

[5] https://onlinelibrary.wiley.com/doi/abs/10.1002/sta4.184
[6] https://cran.r-project.org/web/packages/BRISC/BRISC.pdf

BRISC: Fast NNGP-based spatial bootstrap

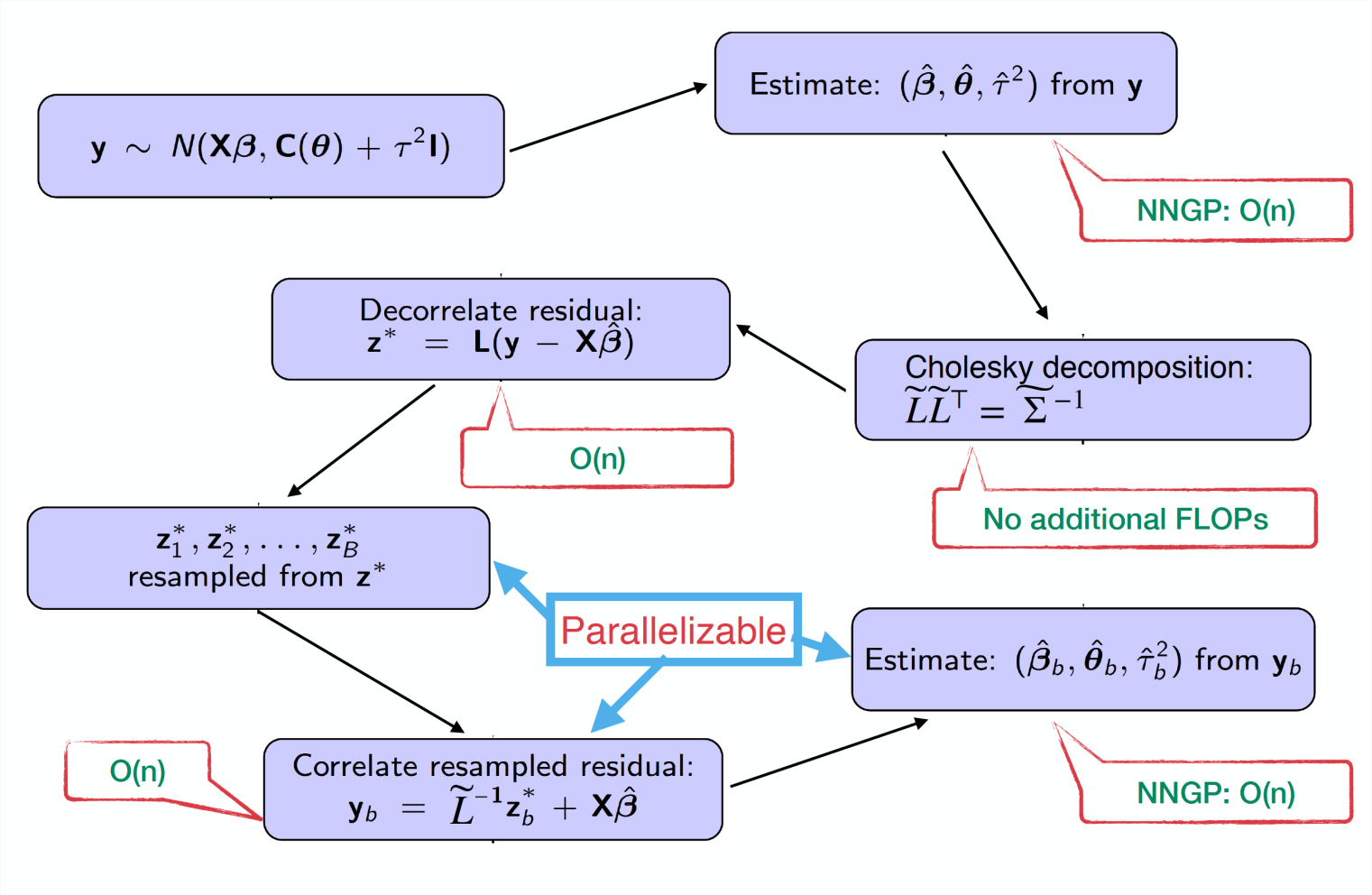


Figure: BRISC algorithm for bootstrapping of all parameters in the spatial linear model. This is implemented in the `BRISC_bootstrap` function of the BRISC package.

Comparing inference from different NNGP models

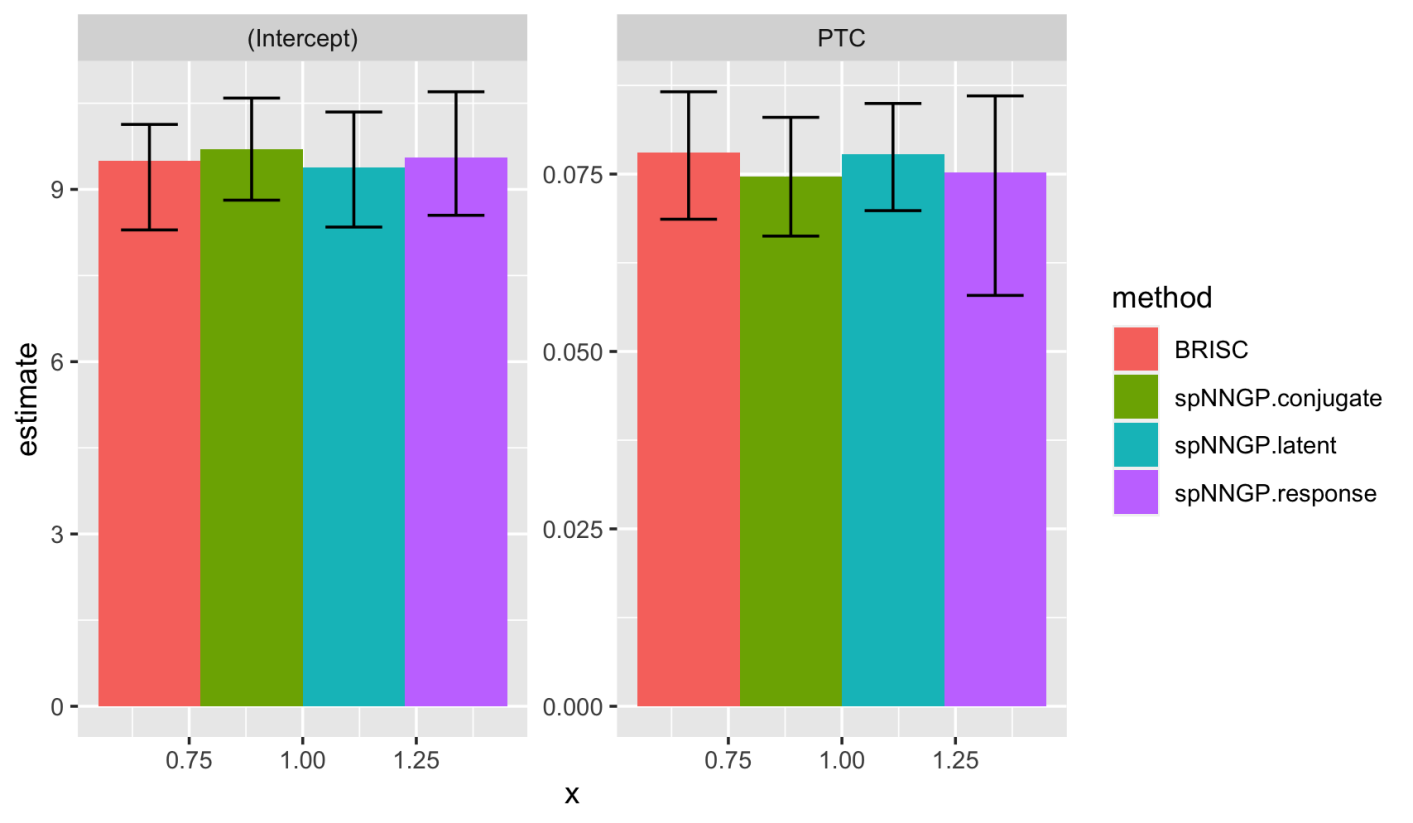


Figure: Comparing estimates of the intercept and the slope on Percent tree cover (PTC) for forest canopy height (FCH) in the Bonanza Creek Experimental Forest dataset using different NNGP models

Fast simulation of large Gaussian Process datasets

- Input: n locations $s_1, \ldots, s_n$ and a GP covariance matrix Σ on these locations
- Generating $y_{sim} \sim N(0, \Sigma)$ requires to store Σ and calculate its Cholesky factor ($O(n^2)$ storage and $O(n^3)$ FLOPs)
- The NNGP covariance matrix $\widetilde{\Sigma} \approx \Sigma$
- `BRISC_simulation` generates $y_{sim} \sim N(0, \widetilde{\Sigma})$
  - Generate $z \sim N(0, I)$
  - Generate sparse Cholesky factor $L = D^{-1/2}(I - A)$ of $\widetilde{\Sigma}^{-1}$
  - Solve sparse triangular system: $Ly_{sim} = z$ to obtain $y_{sim} \sim N(0, \widetilde{\Sigma})$
  - $O(n)$ storage and FLOPs
- Fast approximate large-scale simulation of GP realizations

Fast simulation of large Gaussian Process datasets

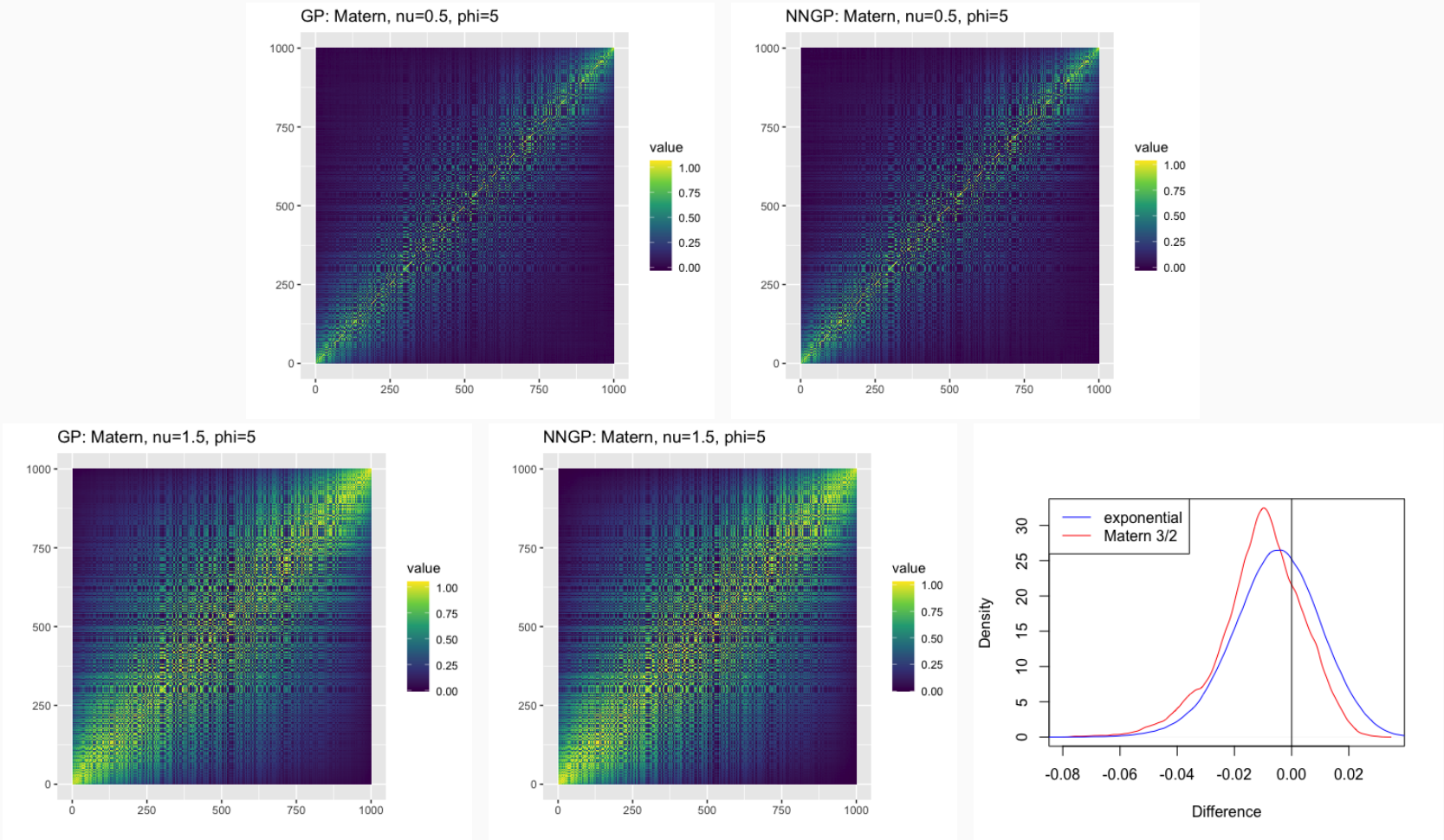


Figure: Simulation of Gaussian random fields using full GP and NNGP. The last figure plots the density of the difference between the sample covariance matrices generated from full GP and NNGP.

Fast simulation of large Gaussian Process datasets

Covariance function	Sample size	NNGP	full GP
exponential	1000	3	12
	2500	6	83
	5000	11	464
	10000	31	NA
	100000	831	NA
$\text{Matérn}_{3/2}$	1000	4	11
	2500	8	81
	5000	18	459
	10000	48	NA
	100000	698	NA

Table: Computation times (in seconds) for simulating 10000 random draws from a full GP and NNGP.[7]

[7] https://arxiv.org/pdf/2102.13299.pdf

Summary of Nearest Neighbor Gaussian Processes

- Sparsity inducing Gaussian process constructed from sparse Cholesky factors based on m nearest neighbors
- Scalability: Storage, inverse and determinant of NNGP covariance matrix are all $O(n)$
- Proper Gaussian process, allows for inference using hierarchical spatial models and predictions at arbitrary spatial resolution
- Closely approximates full GP inference, does not oversmooth like low rank models
- Collapsed and response NNGP models with improved MCMC convergence
- spNNGP package in R for analyzing large spatial data using NNGP models
- Applications in spatial bootstrap, simulation of large Gaussian fields using the BRISC package