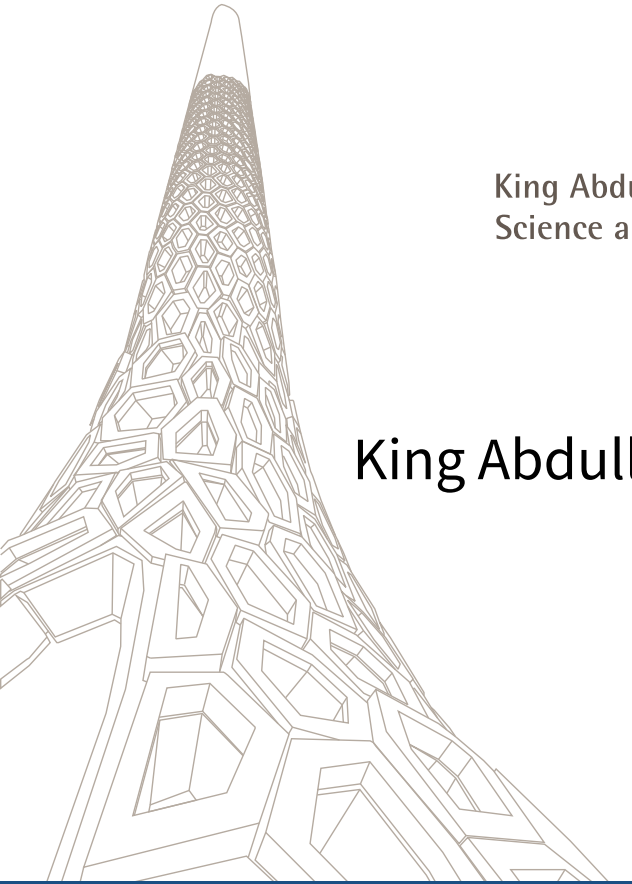
Scalable Bayesian inference for Latent Gaussian Models



King Abdullah University of Science and Technology

جامعة الملك عبدالله للعلوم والتقنية

Håvard Rue
King Abdullah University of Science and Technology
Saudi Arabia

June 2026

Outline

- Importance of GMRFs
- Latent Gaussian Models
- Bayesian Inference
- Key component: (a new) Sparse matrix solver

Outline

- Importance of GMRFs
- Latent Gaussian Models
- Bayesian Inference
- Key component: (a new) Sparse matrix solver

Outline

- Importance of GMRFs
- Latent Gaussian Models
- Bayesian Inference
- Key component: (a new) Sparse matrix solver

Outline

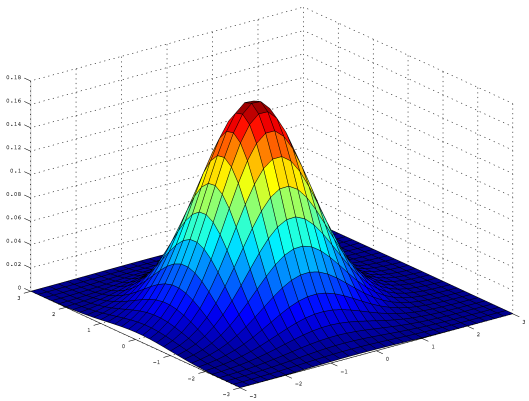
- Importance of GMRFs
- Latent Gaussian Models
- Bayesian Inference
- Key component: (a new) Sparse matrix solver

Multivariate Normal distribution

- Random vector $(x_1, \ldots, x_n)$
- (Scaled and centred) density

$$\pi(\boldsymbol{x}) \propto |\boldsymbol{\Sigma}|^{-1/2} \exp\left(-\frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x}\right)$$

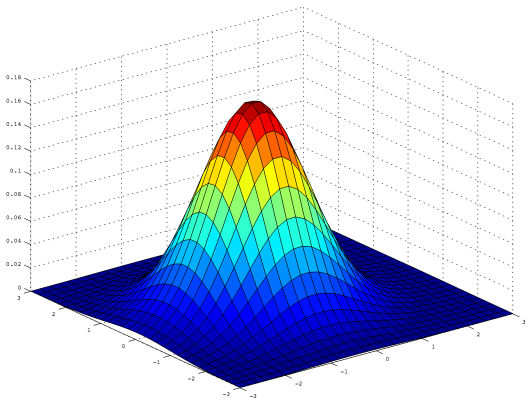
- $\boldsymbol{\Sigma}$ is the (SPD) covariance matrix
- $\Sigma_{ij} = \mathrm{E}(x_i x_j)$
- $\Sigma_{ij} = 0 \Leftrightarrow x_i \perp x_j \; (i \neq j)$



Multivariate Normal distribution

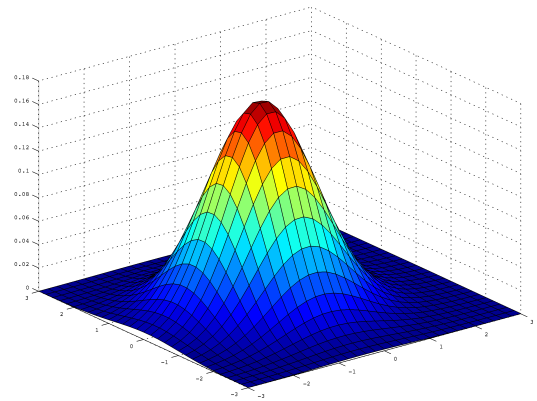
- Random vector $(x_1, \ldots, x_n)$
- (Scaled and centred) density

$$\pi(\boldsymbol{x}) \propto |\boldsymbol{\Sigma}|^{-1/2} \exp\left(-\frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x}\right)$$



- $\boldsymbol{\Sigma}$ is the (SPD) covariance matrix
- $\Sigma_{ij} = \mathrm{E}(x_i x_j)$
- $\Sigma_{ij} = 0 \Leftrightarrow x_i \perp x_j \; (i \neq j)$

Multivariate Normal distribution



- Random vector $(x_1, \ldots, x_n)$
- (Scaled and centred) density

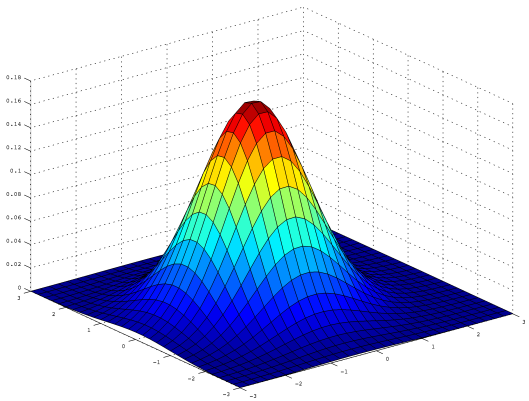
$$\pi(\boldsymbol{x}) \propto |\boldsymbol{\Sigma}|^{-1/2} \exp\left(-\frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x}\right)$$

- $\boldsymbol{\Sigma}$ is the (SPD) covariance matrix
- $\Sigma_{ij} = \mathrm{E}(x_i x_j)$
- $\Sigma_{ij} = 0 \Leftrightarrow x_i \perp x_j \; (i \neq j)$

Multivariate Normal distribution

- Random vector $(x_1, \ldots, x_n)$
- (Scaled and centred) density

$$\pi(\boldsymbol{x}) \propto |\boldsymbol{\Sigma}|^{-1/2} \exp\left(-\frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x}\right)$$

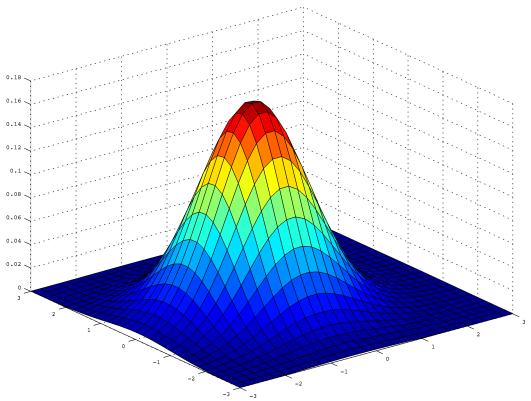


- $\boldsymbol{\Sigma}$ is the (SPD) covariance matrix
- $\Sigma_{ij} = \mathrm{E}(x_i x_j)$
- $\Sigma_{ij} = 0 \Leftrightarrow x_i \perp x_j \ (i \neq j)$

Multivariate Normal distribution

- Random vector $(x_1, \ldots, x_n)$
- (Scaled and centred) density

$$\pi(\boldsymbol{x}) \propto |\boldsymbol{\Sigma}|^{-1/2} \exp\left(-\frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x}\right)$$



- $\boldsymbol{\Sigma}$ is the (SPD) covariance matrix
- $\Sigma_{ij} = \mathrm{E}(x_i x_j)$
- $\Sigma_{ij} = 0 \Leftrightarrow x_i \perp x_j \ (i \neq j)$

Conditional independence and GMRFs

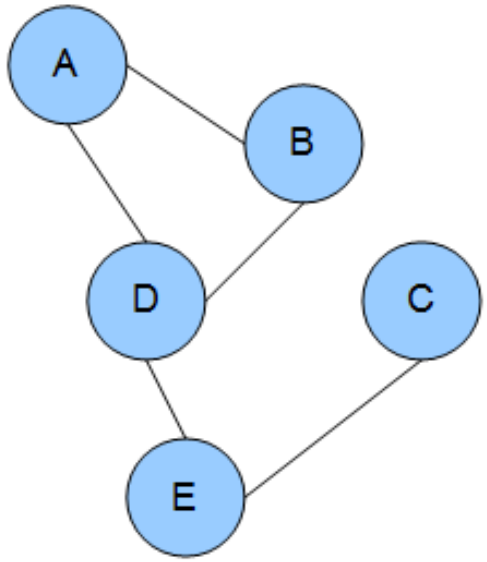
- Conditional independence of x_i and x_j, equals independence in

$$\pi(x_i, x_j \mid \boldsymbol{x}_{-ij})$$

- The result is, with $\boldsymbol{Q} = \boldsymbol{\Sigma}^{-1}$, that

$$Q_{ij} = 0 \Leftrightarrow x_i \perp x_j \mid \boldsymbol{x}_{-ij}$$

- This is potentially very very useful, as
  - If $\boldsymbol{Q}$ is fully connected then $\boldsymbol{\Sigma}$ is dense (Cayley–Hamilton thm)
  - Explain dependence by a sparse $\boldsymbol{Q}$
  - Sparse matrices: faster computations



Conditional independence and GMRFs

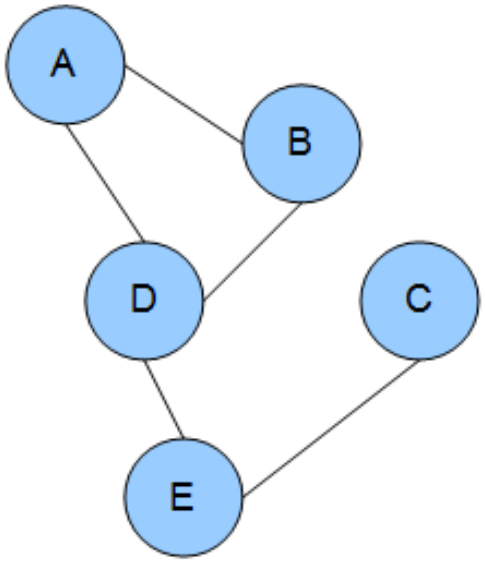
- Conditional independence of x_i and x_j, equals independence in

$$\pi(x_i, x_j \mid \boldsymbol{x}_{-ij})$$

- The result is, with $\boldsymbol{Q} = \boldsymbol{\Sigma}^{-1}$, that

$$Q_{ij} = 0 \Leftrightarrow x_i \perp x_j \mid \boldsymbol{x}_{-ij}$$

- This is potentially very very useful, as
  - If $\boldsymbol{Q}$ is fully connected then $\boldsymbol{\Sigma}$ is dense (Cayley–Hamilton thm)
  - Explain dependence by a sparse $\boldsymbol{Q}$
  - Sparse matrices: faster computations



Conditional independence and GMRFs

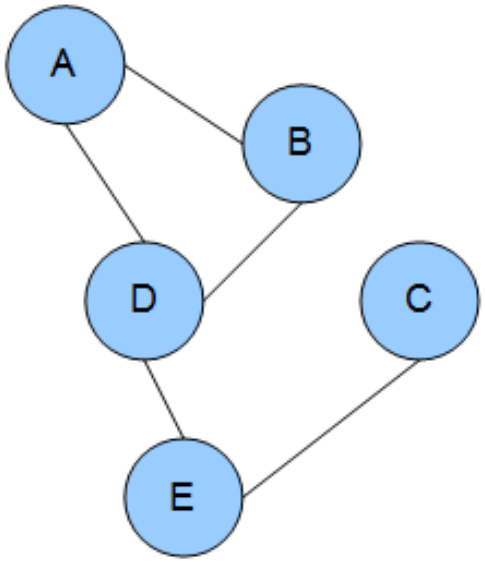
- Conditional independence of x_i and x_j, equals independence in

$$\pi(x_i, x_j \mid \boldsymbol{x}_{-ij})$$

- The result is, with $\boldsymbol{Q} = \boldsymbol{\Sigma}^{-1}$, that

$$Q_{ij} = 0 \Leftrightarrow x_i \perp x_j \mid \boldsymbol{x}_{-ij}$$

- This is potentially very very useful, as
  - If $\boldsymbol{Q}$ is fully connected then $\boldsymbol{\Sigma}$ is dense (Cayley–Hamilton thm)
  - Explain dependence by a sparse $\boldsymbol{Q}$
  - Sparse matrices: faster computations



Conditional independence and GMRFs

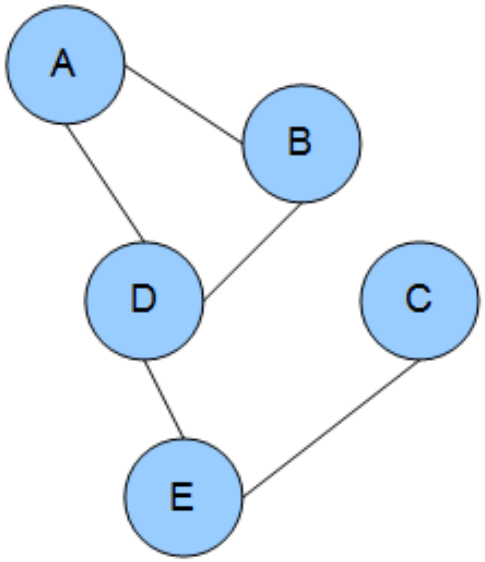
- Conditional independence of x_i and x_j, equals independence in

$$\pi(x_i, x_j \mid \boldsymbol{x}_{-ij})$$

- The result is, with $\boldsymbol{Q} = \boldsymbol{\Sigma}^{-1}$, that

$$Q_{ij} = 0 \Leftrightarrow x_i \perp x_j \mid \boldsymbol{x}_{-ij}$$

- This is potentially very very useful, as
  - If $\boldsymbol{Q}$ is fully connected then $\boldsymbol{\Sigma}$ is dense (Cayley–Hamilton thm)
  - Explain dependence by a sparse $\boldsymbol{Q}$
  - Sparse matrices: faster computations



Conditional independence and GMRFs

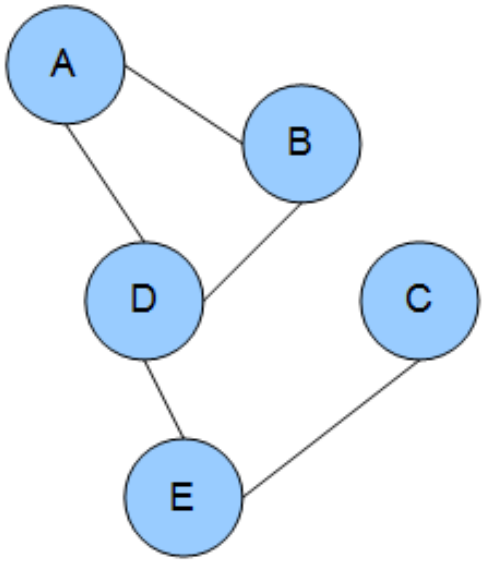
- Conditional independence of x_i and x_j, equals independence in

$$\pi(x_i, x_j \mid \boldsymbol{x}_{-ij})$$

- The result is, with $\boldsymbol{Q} = \boldsymbol{\Sigma}^{-1}$, that

$$Q_{ij} = 0 \Leftrightarrow x_i \perp x_j \mid \boldsymbol{x}_{-ij}$$

- This is potentially very very useful, as
  - If $\boldsymbol{Q}$ is fully connected then $\boldsymbol{\Sigma}$ is dense (Cayley–Hamilton thm)
  - Explain dependence by a sparse $\boldsymbol{Q}$
  - Sparse matrices: faster computations



Conditional independence and GMRFs

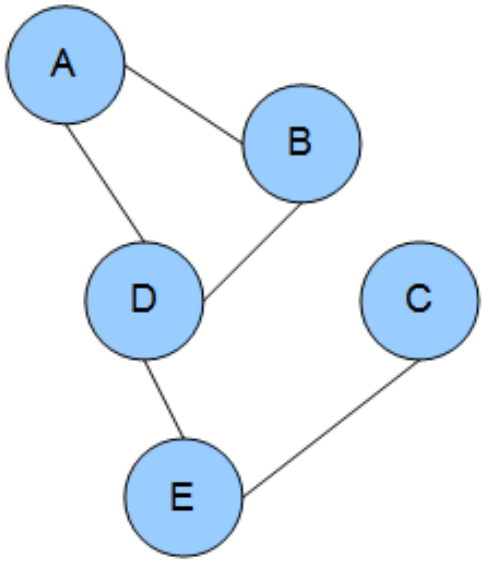
- Conditional independence of x_i and x_j, equals independence in

$$\pi(x_i, x_j \mid \boldsymbol{x}_{-ij})$$

- The result is, with $\boldsymbol{Q} = \boldsymbol{\Sigma}^{-1}$, that

$$Q_{ij} = 0 \Leftrightarrow x_i \perp x_j \mid \boldsymbol{x}_{-ij}$$

- This is potentially very very useful, as
  - If $\boldsymbol{Q}$ is fully connected then $\boldsymbol{\Sigma}$ is dense (Cayley–Hamilton thm)
  - Explain dependence by a sparse $\boldsymbol{Q}$
  - Sparse matrices: faster computations



Example: Markov processes in time

- Auto-regressive process of order 1

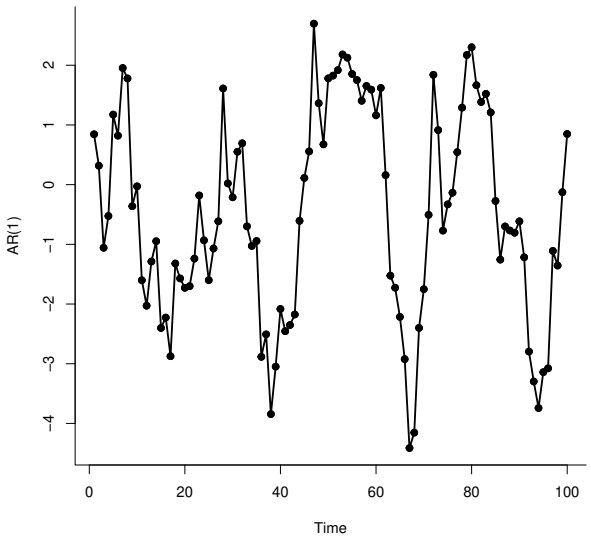
$$x_t = \phi x_{t-1} + \epsilon_t, \quad 0 \le \phi < 1, \quad t = 1, \ldots, n$$

with Gaussian noise ϵ_t

- Linear graph
- $\boldsymbol{Q}$ is tridiagonal (factorisation cost $\mathcal{O}(n)$)
- $\boldsymbol{\Sigma}$ is $\phi^{-|i-j|}$ (exp-correlation function)
- Cont.time: Ornstein-Uhlenbeck process

$$dx_t = -\theta\, x_t\, dt + dW_t,$$

Wiener process W_t



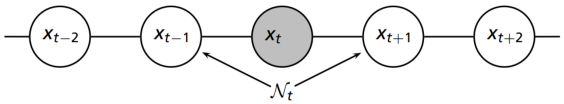
Example: Markov processes in time

- Auto-regressive process of order 1

$$x_t = \phi x_{t-1} + \epsilon_t, \quad 0 \le \phi < 1, \quad t = 1, \ldots, n$$

  with Gaussian noise ϵ_t

- Linear graph



- $\boldsymbol{Q}$ is tridiagonal (factorisation cost $\mathcal{O}(n)$)
- $\boldsymbol{\Sigma}$ is $\phi^{-|i-j|}$ (exp-correlation function)
- Cont.time: Ornstein-Uhlenbeck process

$$dx_t = -\theta\, x_t\, dt + dW_t,$$

  Wiener process W_t

Example: Markov processes in time

- Auto-regressive process of order 1

$$x_t = \phi x_{t-1} + \epsilon_t, \quad 0 \le \phi < 1, \quad t = 1, \ldots, n$$

  with Gaussian noise ϵ_t
- Linear graph
- $\boldsymbol{Q}$ is tridiagonal (factorisation cost $\mathcal{O}(n)$)
- $\boldsymbol{\Sigma}$ is $\phi^{-|i-j|}$ (exp-correlation function)
- Cont.time: Ornstein-Uhlenbeck process

$$dx_t = -\theta\, x_t\, dt + dW_t,$$

  Wiener process W_t

$$\begin{pmatrix} 1 & -\phi & & & & & \\ -\phi & 1+\phi^2 & -\phi & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & & -\phi & 1+\phi^2 & -\phi & \\ & & & & -\phi & 1 \end{pmatrix}$$

Example: Markov processes in time

- Auto-regressive process of order 1

$$x_t = \phi x_{t-1} + \epsilon_t, \quad 0 \le \phi < 1, \quad t = 1, \ldots, n$$

  with Gaussian noise ϵ_t
- Linear graph
- $\boldsymbol{Q}$ is tridiagonal (factorisation cost $\mathcal{O}(n)$)
- $\boldsymbol{\Sigma}$ is $\phi^{-|i-j|}$ (exp-correlation function)
- Cont.time: Ornstein-Uhlenbeck process

$$dx_t = -\theta\, x_t\, dt + dW_t,$$

  Wiener process W_t

$$\begin{pmatrix} 1 & \phi & \phi^2 & \phi^3 & \phi^4 & \phi^5 & \phi^6 \\ \phi & 1 & \phi & \phi^2 & \phi^3 & \phi^4 & \phi^5 \\ \phi^2 & \phi & 1 & \phi & \phi^2 & \phi^3 & \phi^4 \\ \phi^3 & \phi^2 & \phi & 1 & \phi & \phi^2 & \phi^3 \\ \phi^4 & \phi^3 & \phi^2 & \phi & 1 & \phi & \phi^2 \\ \phi^5 & \phi^4 & \phi^3 & \phi^2 & \phi & 1 & \phi \\ \phi^6 & \phi^5 & \phi^4 & \phi^3 & \phi^2 & \phi & 1 \end{pmatrix}$$

Example: Markov processes in time

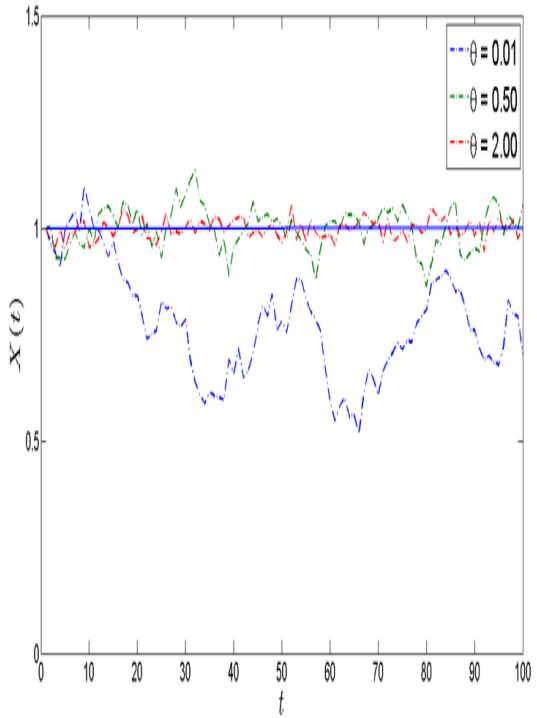
- Auto-regressive process of order 1

$$x_t = \phi x_{t-1} + \epsilon_t, \quad 0 \le \phi < 1, \quad t = 1, \ldots, n$$

  with Gaussian noise ϵ_t
- Linear graph
- $\boldsymbol{Q}$ is tridiagonal (factorisation cost $\mathcal{O}(n)$)
- $\boldsymbol{\Sigma}$ is $\phi^{-|i-j|}$ (exp-correlation function)
- Cont.time: Ornstein-Uhlenbeck process

$$dx_t = -\theta\, x_t\, dt + dW_t,$$

  Wiener process W_t



How does correlation work?

- Let

$$x_1 = z_1 + \mu$$

$$x_2 = z_2 + \mu$$

- z_1, z_2, μ are independent zero mean Gaussian's
- $\text{Var}(\mu) = \sigma^2$, $\text{Var}(z_i) = 1$
- Then

$$\text{Corr}(x_1, x_2) = \frac{\sigma^2}{\sigma^2 + 1}$$

- This is an constructive/intuitive way to introduce correlation (which is a 'consequence' of the construction)

How does correlation work?

- Let

$$x_1 = z_1 + \mu$$

$$x_2 = z_2 + \mu$$

- z_1, z_2, μ are independent zero mean Gaussian's
- $\text{Var}(\mu) = \sigma^2$, $\text{Var}(z_i) = 1$
- Then

$$\text{Corr}(x_1, x_2) = \frac{\sigma^2}{\sigma^2 + 1}$$

- This is an constructive/intuitive way to introduce correlation (which is a 'consequence' of the construction)

How does correlation work?

- Let

$$x_1 = z_1 + \mu$$

$$x_2 = z_2 + \mu$$

- z_1, z_2, μ are independent zero mean Gaussian's
- $\text{Var}(\mu) = \sigma^2$, $\text{Var}(z_i) = 1$
- Then

$$\text{Corr}(x_1, x_2) = \frac{\sigma^2}{\sigma^2 + 1}$$

- This is an constructive/intuitive way to introduce correlation (which is a 'consequence' of the construction)

How does correlation work?

- Let

$$x_1 = z_1 + \mu$$

$$x_2 = z_2 + \mu$$

- z_1, z_2, μ are independent zero mean Gaussian's
- $\text{Var}(\mu) = \sigma^2$, $\text{Var}(z_i) = 1$
- Then

$$\text{Corr}(x_1, x_2) = \frac{\sigma^2}{\sigma^2 + 1}$$

- This is an constructive/intuitive way to introduce correlation (which is a 'consequence' of the construction)

How does correlation work?

- Let

$$x_1 = z_1 + \mu$$

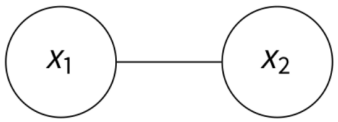
$$x_2 = z_2 + \mu$$

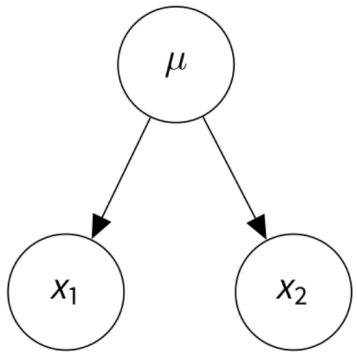
- z_1, z_2, μ are independent zero mean Gaussian's
- $\text{Var}(\mu) = \sigma^2$, $\text{Var}(z_i) = 1$
- Then

$$\text{Corr}(x_1, x_2) = \frac{\sigma^2}{\sigma^2 + 1}$$

- This is an constructive/intuitive way to introduce correlation (which is a 'consequence' of the construction)

How this influence sparsity?

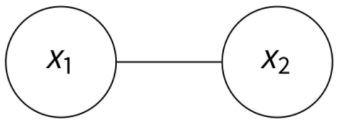


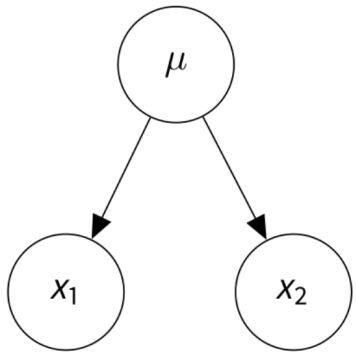


$$Q_{x_1,x_2} = \begin{pmatrix} \times & \times \\ \times & \times \end{pmatrix}$$

$$Q_{x_1,x_2,\mu} = \begin{pmatrix} \times & & \times \\ & \times & \times \\ \times & \times & \times \end{pmatrix}$$

How this influence sparsity?

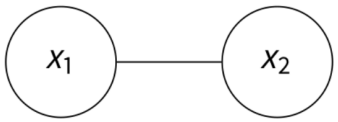


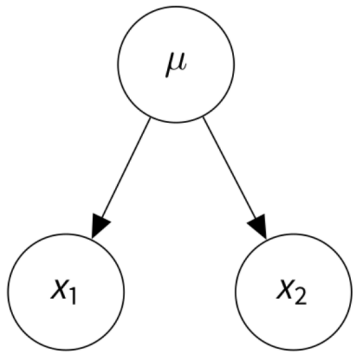


$$\boldsymbol{Q}_{x_1,x_2} = \begin{pmatrix} \times & \times \\ \times & \times \end{pmatrix}$$

$$\boldsymbol{Q}_{x_1,x_2,\mu} = \begin{pmatrix} \times & & \times \\ & \times & \times \\ \times & \times & \times \end{pmatrix}$$

How this influence sparsity?

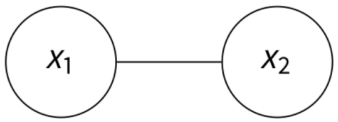


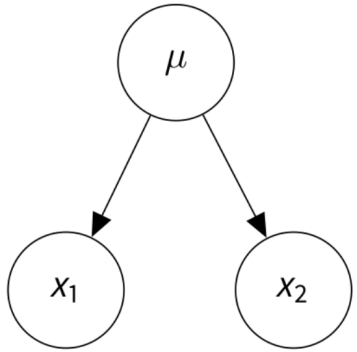


$$\boldsymbol{Q}_{x_1,x_2} = \begin{pmatrix} \times & \times \\ \times & \times \end{pmatrix}$$

$$\boldsymbol{Q}_{x_1,x_2,\mu} = \begin{pmatrix} \times & & \times \\ & \times & \times \\ \times & \times & \times \end{pmatrix}$$

How this influence sparsity?





$$\boldsymbol{Q}_{x_1,x_2} = \begin{pmatrix} \times & \times \\ \times & \times \end{pmatrix}$$

$$\boldsymbol{Q}_{x_1,x_2,\mu} = \begin{pmatrix} \times & & \times \\ & \times & \times \\ \times & \times & \times \end{pmatrix}$$

What are good "operations" for GMRFs?

Conditioning $\boldsymbol{x}_1|\boldsymbol{x}_2$

$$\boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{12}^T & \boldsymbol{\Sigma}_{22} \end{pmatrix} \qquad \boldsymbol{Q} = \begin{pmatrix} \boldsymbol{Q}_{11} & \boldsymbol{Q}_{12} \\ \boldsymbol{Q}_{12}^T & \boldsymbol{Q}_{22} \end{pmatrix}$$

$$\text{Var}(\boldsymbol{x}_1|\boldsymbol{x}_2) = \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{12}^T \qquad \text{Prec}(\boldsymbol{x}_1|\boldsymbol{x}_2) = \boldsymbol{Q}_{11}$$

What are good "operations" for GMRFs?

Updating/Learning:

$$\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{x}}^{-1})$$

then we observe

$$\boldsymbol{y}|\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{x}, \boldsymbol{Q}_{\boldsymbol{y}}^{-1})$$

$$\text{Var}(\boldsymbol{x}|\boldsymbol{y}) = \left(\boldsymbol{\Sigma}_{\boldsymbol{x}}^{-1} + \boldsymbol{\Sigma}_{\boldsymbol{y}}^{-1}\right)^{-1} \qquad \text{Prec}(\boldsymbol{x}|\boldsymbol{y}) = \boldsymbol{Q}_{\boldsymbol{x}} + \boldsymbol{Q}_{\boldsymbol{y}}$$

What are good "operations" for GMRFs?

Updating/Learning:

$$\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{x}}^{-1})$$

then we observe

$$\boldsymbol{y}|\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{x}, \boldsymbol{Q}_{\boldsymbol{y}}^{-1})$$

$$\text{Var}(\boldsymbol{x}|\boldsymbol{y}) = \left(\boldsymbol{\Sigma}_{\boldsymbol{x}}^{-1} + \boldsymbol{\Sigma}_{\boldsymbol{y}}^{-1}\right)^{-1} \qquad \text{Prec}(\boldsymbol{x}|\boldsymbol{y}) = \boldsymbol{Q}_{\boldsymbol{x}} + \boldsymbol{Q}_{\boldsymbol{y}}$$

What are good "operations" for GMRFs?

Updating/Learning:

$$\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{x}}^{-1})$$

then we observe

$$\boldsymbol{y}|\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{x}, \boldsymbol{Q}_{\boldsymbol{y}}^{-1})$$

$$\text{Var}(\boldsymbol{x}|\boldsymbol{y}) = \left(\boldsymbol{\Sigma}_{\boldsymbol{x}}^{-1} + \boldsymbol{\Sigma}_{\boldsymbol{y}}^{-1}\right)^{-1} \qquad \text{Prec}(\boldsymbol{x}|\boldsymbol{y}) = \boldsymbol{Q}_{\boldsymbol{x}} + \boldsymbol{Q}_{\boldsymbol{y}}$$

What are good "operations" for GMRFs?

Distribution of sums:

$$\eta = \boldsymbol{x} + \boldsymbol{y} + \boldsymbol{z}$$

where $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{x}}^{-1})$, $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{y}}^{-1})$ and $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{z}}^{-1})$.

$$\boldsymbol{Q}_\eta = \left(\boldsymbol{Q}_{\boldsymbol{x}}^{-1} + \boldsymbol{Q}_{\boldsymbol{y}}^{-1} + \boldsymbol{Q}_{\boldsymbol{z}}^{-1}\right)^{-1}$$

$\boldsymbol{Q}_\eta$ is (in general) dense and it is costly to update $\boldsymbol{Q}_\eta$

The covariance formulation is easy

$$\boldsymbol{\Sigma}_\eta = \boldsymbol{\Sigma}_{\boldsymbol{x}} + \boldsymbol{\Sigma}_{\boldsymbol{y}} + \boldsymbol{\Sigma}_{\boldsymbol{z}}$$

Marginalisation destroy sparsity...

What are good "operations" for GMRFs?

Distribution of sums:

$$\eta = \boldsymbol{x} + \boldsymbol{y} + \boldsymbol{z}$$

where $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{x}}^{-1})$, $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{y}}^{-1})$ and $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{z}}^{-1})$.

$$\boldsymbol{Q}_\eta = \left(\boldsymbol{Q}_{\boldsymbol{x}}^{-1} + \boldsymbol{Q}_{\boldsymbol{y}}^{-1} + \boldsymbol{Q}_{\boldsymbol{z}}^{-1}\right)^{-1}$$

$\boldsymbol{Q}_\eta$ is (in general) dense and it is costly to update $\boldsymbol{Q}_\eta$

The covariance formulation is easy

$$\boldsymbol{\Sigma}_\eta = \boldsymbol{\Sigma}_{\boldsymbol{x}} + \boldsymbol{\Sigma}_{\boldsymbol{y}} + \boldsymbol{\Sigma}_{\boldsymbol{z}}$$

Marginalisation destroy sparsity...

What are good "operations" for GMRFs?

Distribution of sums:

$$\eta = \boldsymbol{x} + \boldsymbol{y} + \boldsymbol{z}$$

where $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{x}}^{-1})$, $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{y}}^{-1})$ and $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{z}}^{-1})$.

$$\boldsymbol{Q}_{\eta} = \left(\boldsymbol{Q}_{\boldsymbol{x}}^{-1} + \boldsymbol{Q}_{\boldsymbol{y}}^{-1} + \boldsymbol{Q}_{\boldsymbol{z}}^{-1}\right)^{-1}$$

$\boldsymbol{Q}_{\eta}$ is (in general) dense and it is costly to update $\boldsymbol{Q}_{\eta}$

The covariance formulation is easy

$$\boldsymbol{\Sigma}_{\eta} = \boldsymbol{\Sigma}_{\boldsymbol{x}} + \boldsymbol{\Sigma}_{\boldsymbol{y}} + \boldsymbol{\Sigma}_{\boldsymbol{z}}$$

Marginalisation destroy sparsity...

What are good "operations" for GMRFs?

Distribution of sums:

$$\eta = \boldsymbol{x} + \boldsymbol{y} + \boldsymbol{z}$$

where $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{x}}^{-1})$, $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{y}}^{-1})$ and $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{z}}^{-1})$.

$$\boldsymbol{Q}_\eta = \left(\boldsymbol{Q}_{\boldsymbol{x}}^{-1} + \boldsymbol{Q}_{\boldsymbol{y}}^{-1} + \boldsymbol{Q}_{\boldsymbol{z}}^{-1}\right)^{-1}$$

$\boldsymbol{Q}_\eta$ is (in general) dense and it is costly to update $\boldsymbol{Q}_\eta$

The covariance formulation is easy

$$\boldsymbol{\Sigma}_\eta = \boldsymbol{\Sigma}_{\boldsymbol{x}} + \boldsymbol{\Sigma}_{\boldsymbol{y}} + \boldsymbol{\Sigma}_{\boldsymbol{z}}$$

Marginalisation destroy sparsity...

What are good "operations" for GMRFs?

Distribution of sums:

$$\boldsymbol{\eta} = \boldsymbol{x} + \boldsymbol{y} + \boldsymbol{z}$$

where $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{x}}^{-1})$, $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{y}}^{-1})$ and $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{z}}^{-1})$.

- Using "ϵ" with high precision τ

$$\boldsymbol{\eta} = \boldsymbol{x} + \boldsymbol{y} + \boldsymbol{z} + \epsilon$$

- Then

$$\text{Var}\begin{pmatrix} \eta \\ x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \tau I & -\tau I & -\tau I & -\tau I \\ & Q_x + \tau I & \tau I & \tau I \\ & & Q_y + \tau I & \tau I \\ & & & Q_z + \tau I \end{pmatrix}$$

- Good practical solution unless choice of τ becomes an issue

What are good "operations" for GMRFs?

Distribution of sums:

$$\eta = x + y + z$$

where $x \sim \mathcal{N}(0, Q_x^{-1})$, $y \sim \mathcal{N}(0, Q_y^{-1})$ and $z \sim \mathcal{N}(0, Q_z^{-1})$.

- Using "ϵ" with high precision τ

$$\eta = x + y + z + \epsilon$$

- Then

$$\text{Var}\begin{pmatrix} \eta \\ x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \tau I & -\tau I & -\tau I & -\tau I \\ & Q_x + \tau I & \tau I & \tau I \\ & & Q_y + \tau I & \tau I \\ & & & Q_z + \tau I \end{pmatrix}$$

- Good practical solution unless choice of τ becomes an issue

What are good "operations" for GMRFs?

Distribution of sums:

$$\boldsymbol{\eta} = \boldsymbol{x} + \boldsymbol{y} + \boldsymbol{z}$$

where $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{x}}^{-1})$, $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{y}}^{-1})$ and $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{z}}^{-1})$.

- Using "ϵ" with high precision τ

$$\boldsymbol{\eta} = \boldsymbol{x} + \boldsymbol{y} + \boldsymbol{z} + \epsilon$$

- Then

$$\text{Var}\begin{pmatrix} \boldsymbol{\eta} \\ \boldsymbol{x} \\ \boldsymbol{y} \\ \boldsymbol{z} \end{pmatrix} = \begin{pmatrix} \tau \boldsymbol{I} & -\tau \boldsymbol{I} & -\tau \boldsymbol{I} & -\tau \boldsymbol{I} \\ & \boldsymbol{Q}_{\boldsymbol{x}} + \tau \boldsymbol{I} & \tau \boldsymbol{I} & \tau \boldsymbol{I} \\ & & \boldsymbol{Q}_{\boldsymbol{y}} + \tau \boldsymbol{I} & \tau \boldsymbol{I} \\ & & & \boldsymbol{Q}_{\boldsymbol{z}} + \tau \boldsymbol{I} \end{pmatrix}$$

- Good practical solution unless choice of τ becomes an issue

What are good "operations" for GMRFs?

Distribution of sums:

$$\eta = \boldsymbol{x} + \boldsymbol{y} + \boldsymbol{z}$$

where $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_x^{-1})$, $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_y^{-1})$ and $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_z^{-1})$.

- Using cummulative sums

$$\begin{pmatrix} \widetilde{\boldsymbol{x}} \\ \widetilde{\boldsymbol{y}} \\ \widetilde{\boldsymbol{z}} \end{pmatrix} = \begin{pmatrix} \boldsymbol{x} + \boldsymbol{y} + \boldsymbol{z} \\ \boldsymbol{y} + \boldsymbol{z} \\ \boldsymbol{z} \end{pmatrix}$$

- Then

$$\text{Var} \begin{pmatrix} \widetilde{\boldsymbol{x}} \\ \widetilde{\boldsymbol{y}} \\ \widetilde{\boldsymbol{z}} \end{pmatrix} = \begin{pmatrix} \boldsymbol{Q}_x & -\boldsymbol{Q}_x & \\ & \boldsymbol{Q}_y + \boldsymbol{Q}_x & -\boldsymbol{Q}_y \\ & & \boldsymbol{Q}_z + \boldsymbol{Q}_y \end{pmatrix}$$

- More efficient, but no access to $\boldsymbol{x}$ and $\boldsymbol{y}$ directly.

What are good "operations" for GMRFs?

Distribution of sums:

$$\eta = \boldsymbol{x} + \boldsymbol{y} + \boldsymbol{z}$$

where $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_x^{-1})$, $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_y^{-1})$ and $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_z^{-1})$.

- Using cummulative sums

$$\begin{pmatrix} \widetilde{\boldsymbol{x}} \\ \widetilde{\boldsymbol{y}} \\ \widetilde{\boldsymbol{z}} \end{pmatrix} = \begin{pmatrix} \boldsymbol{x} + \boldsymbol{y} + \boldsymbol{z} \\ \boldsymbol{y} + \boldsymbol{z} \\ \boldsymbol{z} \end{pmatrix}$$

- Then

$$\text{Var} \begin{pmatrix} \widetilde{\boldsymbol{x}} \\ \widetilde{\boldsymbol{y}} \\ \widetilde{\boldsymbol{z}} \end{pmatrix} = \begin{pmatrix} \boldsymbol{Q}_x & -\boldsymbol{Q}_x & \\ -\boldsymbol{Q}_x & \boldsymbol{Q}_y + \boldsymbol{Q}_x & -\boldsymbol{Q}_y \\ & -\boldsymbol{Q}_y & \boldsymbol{Q}_z + \boldsymbol{Q}_y \end{pmatrix}$$

- More efficient, but no access to $\boldsymbol{x}$ and $\boldsymbol{y}$ directly.

What are good "operations" for GMRFs?

Distribution of sums:

$$\eta = \boldsymbol{x} + \boldsymbol{y} + \boldsymbol{z}$$

where $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{x}}^{-1})$, $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{y}}^{-1})$ and $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{z}}^{-1})$.

- Using cummulative sums

$$\begin{pmatrix} \widetilde{\boldsymbol{x}} \\ \widetilde{\boldsymbol{y}} \\ \widetilde{\boldsymbol{z}} \end{pmatrix} = \begin{pmatrix} \boldsymbol{x} + \boldsymbol{y} + \boldsymbol{z} \\ \boldsymbol{y} + \boldsymbol{z} \\ \boldsymbol{z} \end{pmatrix}$$

- Then

$$\text{Var} \begin{pmatrix} \widetilde{\boldsymbol{x}} \\ \widetilde{\boldsymbol{y}} \\ \widetilde{\boldsymbol{z}} \end{pmatrix} = \begin{pmatrix} \boldsymbol{Q}_{\boldsymbol{x}} & -\boldsymbol{Q}_{\boldsymbol{x}} & \\ -\boldsymbol{Q}_{\boldsymbol{x}} & \boldsymbol{Q}_{\boldsymbol{y}} + \boldsymbol{Q}_{\boldsymbol{x}} & -\boldsymbol{Q}_{\boldsymbol{y}} \\ & -\boldsymbol{Q}_{\boldsymbol{y}} & \boldsymbol{Q}_{\boldsymbol{z}} + \boldsymbol{Q}_{\boldsymbol{y}} \end{pmatrix}$$

- More efficient, but no access to $\boldsymbol{x}$ and $\boldsymbol{y}$ directly.

Additive regression models/GLM++

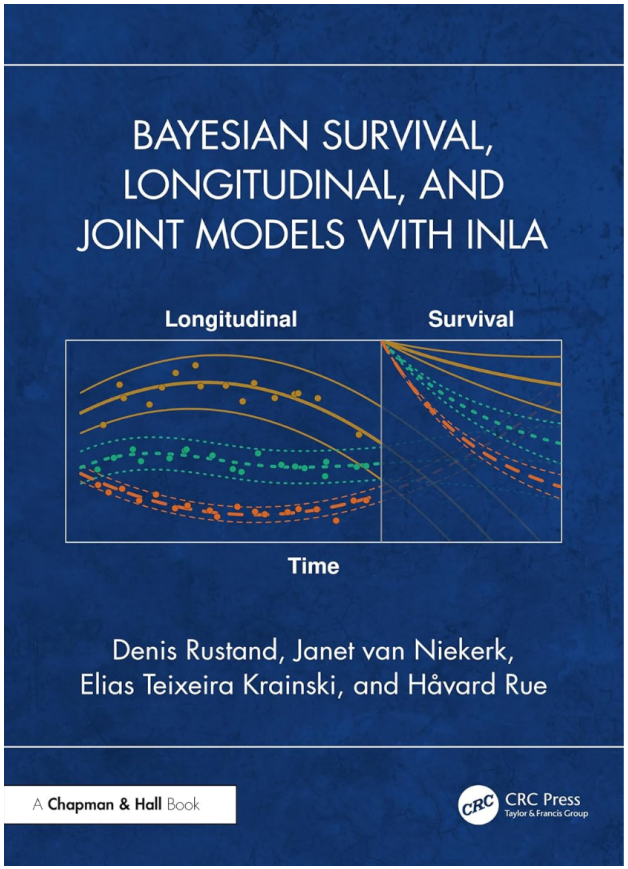
- Linear predictor

$$\eta_i = \mu + \sum_j \beta_j z_{ij} + \boldsymbol{f}_{1i} + \boldsymbol{f}_{2i} + \cdots$$

with observations

$$y_i | \ldots \sim \pi(y_i | \eta_i, \ldots)$$

- $\boldsymbol{f}_k$ is a Gaussian process, think times-series, spatial-model, spline, some kind of dependent 'random effect', measurement error model, etc...



Additive regression models/GLM++

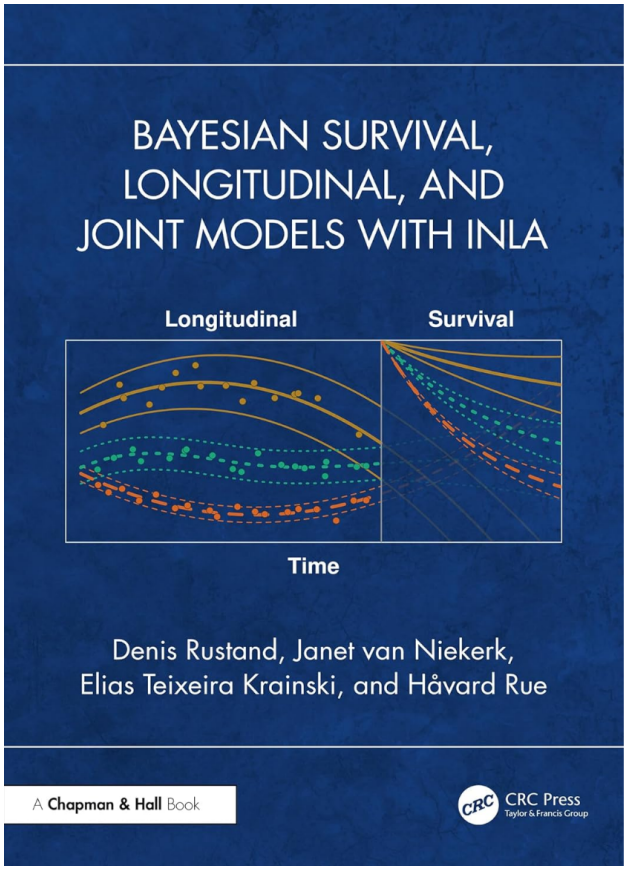
- Linear predictor

$$\eta_i = \mu + \sum_j \beta_j z_{ij} + \boldsymbol{f}_{1i} + \boldsymbol{f}_{2i} + \cdots$$

with observations

$$y_i | \ldots \sim \pi(y_i | \eta_i, \ldots)$$

- $\boldsymbol{f}_k$ is a Gaussian process, think times-series, spatial-model, spline, some kind of dependent 'random effect', measurement error model, etc...



Key observation

- If each $\boldsymbol{f}_k$ is a GMRF, then

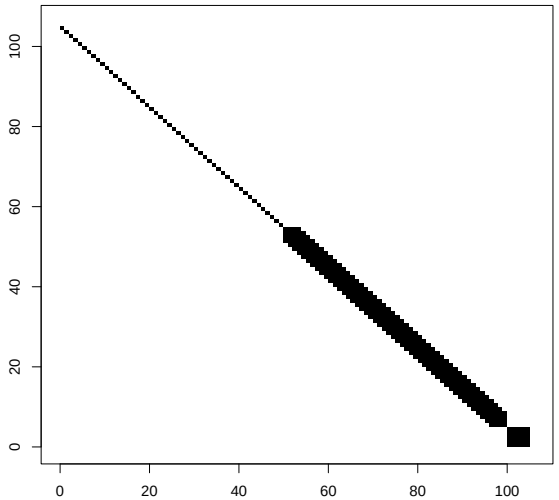
$$\boldsymbol{x} = (\beta_1, \beta_2, \ldots, \boldsymbol{f}_1, \boldsymbol{f}_2, \ldots)$$

  is also a GMRF with precision matrix $\boldsymbol{Q}$

- Let $\eta = \boldsymbol{A}\boldsymbol{x}$. Assume Gaussian data with unit variance, then $\boldsymbol{x}|\boldsymbol{y}$ is a GMRF with precision matrix

$$\boldsymbol{Q} + \boldsymbol{A}^T\boldsymbol{A}$$

- Non-Gaussian data: "sequence of Gaussian-data"



Key observation

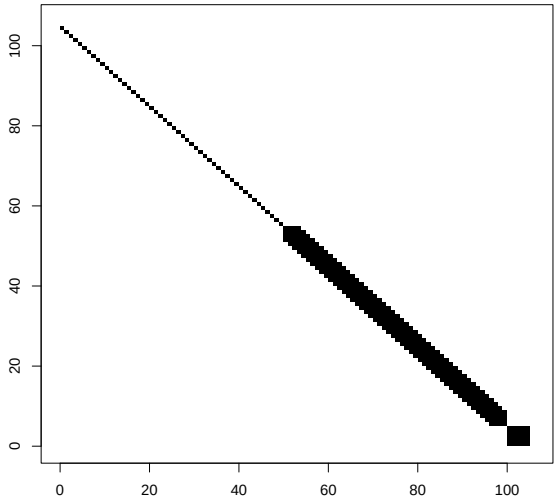
- If each $\boldsymbol{f}_k$ is a GMRF, then

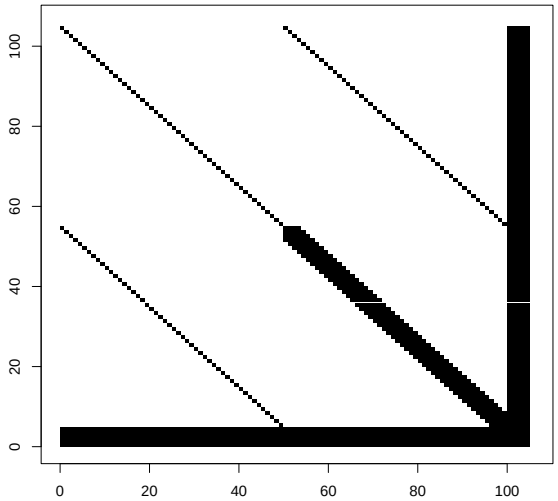
$$\boldsymbol{x} = (\beta_1, \beta_2, \ldots, \boldsymbol{f}_1, \boldsymbol{f}_2, \ldots)$$

  is also a GMRF with precision matrix $\boldsymbol{Q}$
- Let $\boldsymbol{\eta} = \boldsymbol{A}\boldsymbol{x}$. Assume Gaussian data with unit variance, then $\boldsymbol{x}|\boldsymbol{y}$ is a GMRF with precision matrix

$$\boldsymbol{Q} + \boldsymbol{A}^T\boldsymbol{A}$$

- Non-Gaussian data: "sequence of Gaussian-data"





Key observation

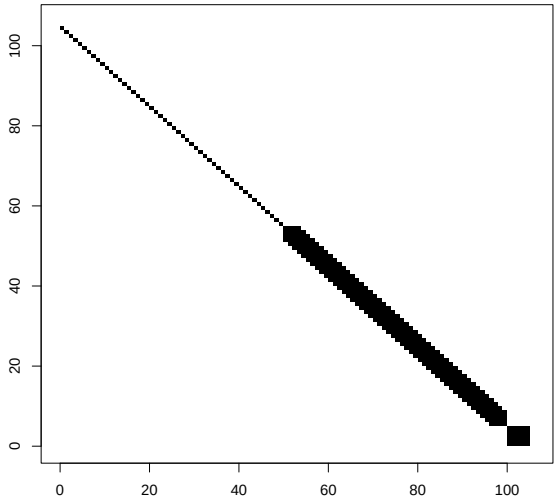
- If each $\boldsymbol{f}_k$ is a GMRF, then

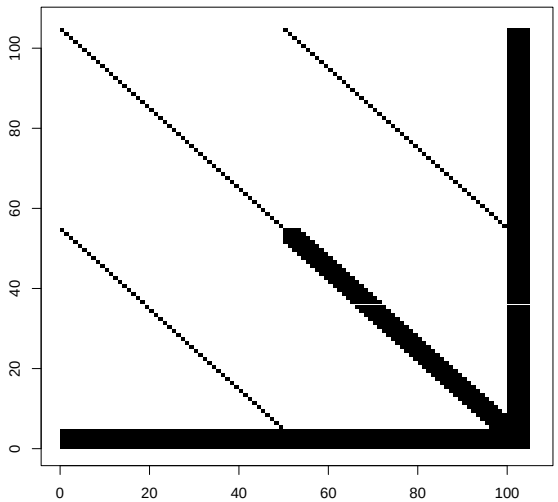
$$\boldsymbol{x} = (\beta_1, \beta_2, \ldots, \boldsymbol{f}_1, \boldsymbol{f}_2, \ldots)$$

  is also a GMRF with precision matrix $\boldsymbol{Q}$
- Let $\boldsymbol{\eta} = \boldsymbol{A}\boldsymbol{x}$. Assume Gaussian data with unit variance, then $\boldsymbol{x}|\boldsymbol{y}$ is a GMRF with precision matrix

$$\boldsymbol{Q} + \boldsymbol{A}^T\boldsymbol{A}$$

- Non-Gaussian data: “sequence of Gaussian-data”





Keep it in the family!

GMRFs is closed

- sums
- conditioning

iff done right: no marginalisation!

Keep it in the family!

GMRFs is closed

- sums
- conditioning

iff done right: no marginalisation!

Keep it in the family!

GMRFs is closed

- sums
- conditioning

iff done right: no marginalisation!

Keep it in the family!

GMRFs is closed

- sums
- conditioning

iff done right: no marginalisation!

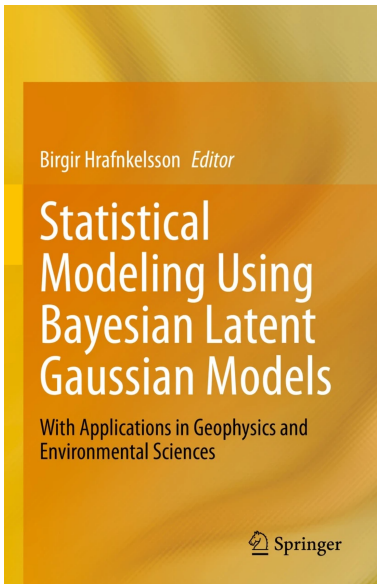
Bayesian inference: Latent Gaussian models (LGMs)

The most important class of models...

- Conditionally independent data $y_i|\eta_i, \theta$
- Latent

$$\eta_i = \sum_j \beta_j c_{ij} + \sum_k f_{ki}$$

- Hyperparameters θ (latent and/or likelihood)
- Each component is a GMRF (or a "small" dense)
- Covariance is a property, not a defining quantity



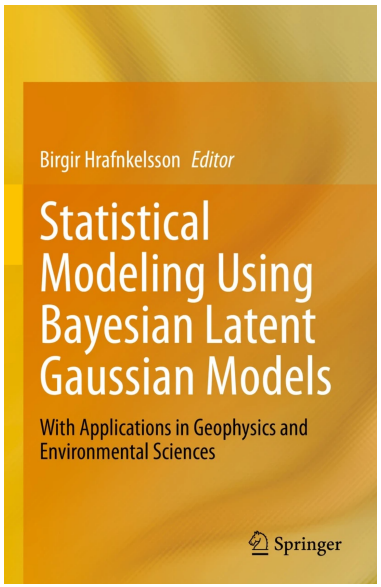
Bayesian inference: Latent Gaussian models (LGMs)

The most important class of models...

- Conditionally independent data $y_i|\eta_i, \theta$
- Latent

$$\eta_i = \sum_j \beta_j c_{ij} + \sum_k f_{ki}$$

- Hyperparameters θ (latent and/or likelihood)
- Each component is a GMRF (or a "small" dense)
- Covariance is a property, not a defining quantity



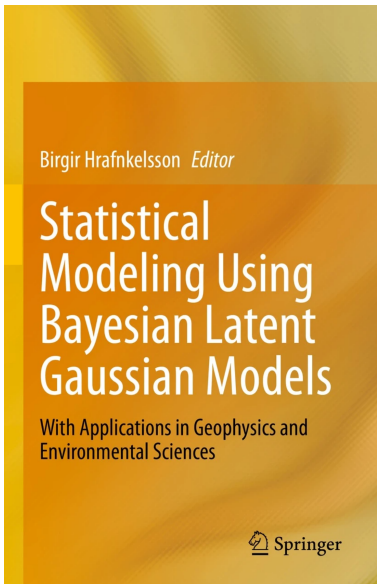
Bayesian inference: Latent Gaussian models (LGMs)

The most important class of models...

- Conditionally independent data $y_i|\eta_i, \boldsymbol{\theta}$
- Latent

$$\eta_i = \sum_j \beta_j c_{ij} + \sum_k f_{ki}$$

- Hyperparameters $\boldsymbol{\theta}$ (latent and/or likelihood)
- Each component is a GMRF (or a "small" dense)
- Covariance is a property, not a defining quantity



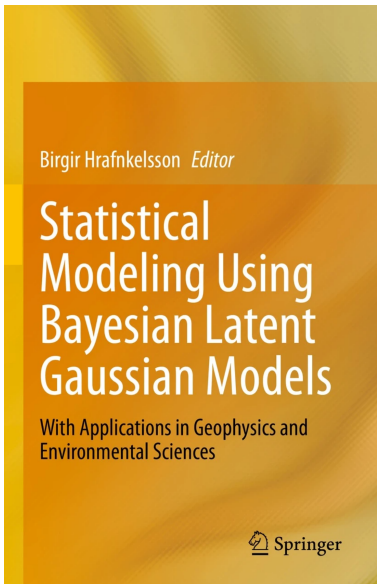
Bayesian inference: Latent Gaussian models (LGMs)

The most important class of models...

- Conditionally independent data $y_i|\eta_i, \theta$
- Latent

$$\eta_i = \sum_j \beta_j c_{ij} + \sum_k f_{ki}$$

- Hyperparameters θ (latent and/or likelihood)
- Each component is a GMRF (or a "small" dense)
- Covariance is a property, not a defining quantity



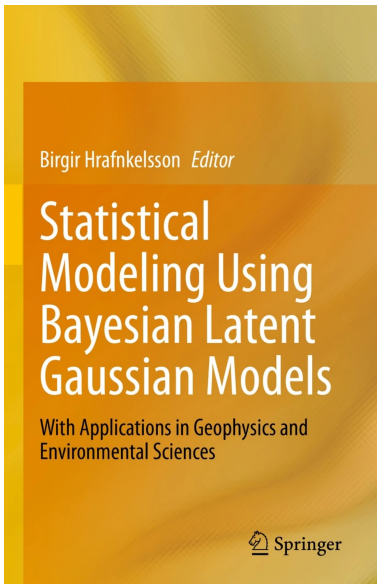
Bayesian inference: Latent Gaussian models (LGMs)

The most important class of models...

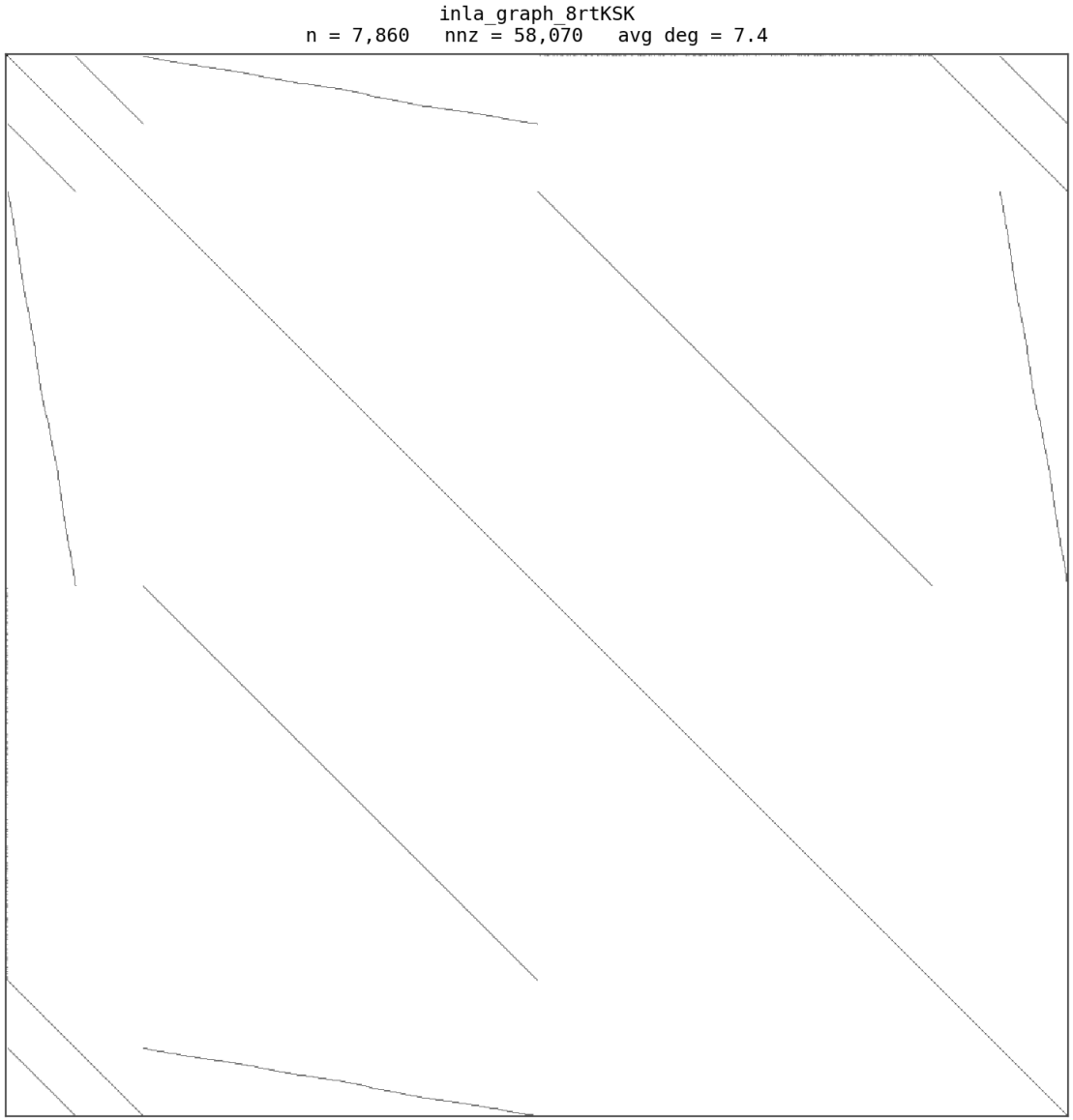
- Conditionally independent data $y_i|\eta_i, \boldsymbol{\theta}$
- Latent

$$\eta_i = \sum_j \beta_j c_{ij} + \sum_k f_{ki}$$

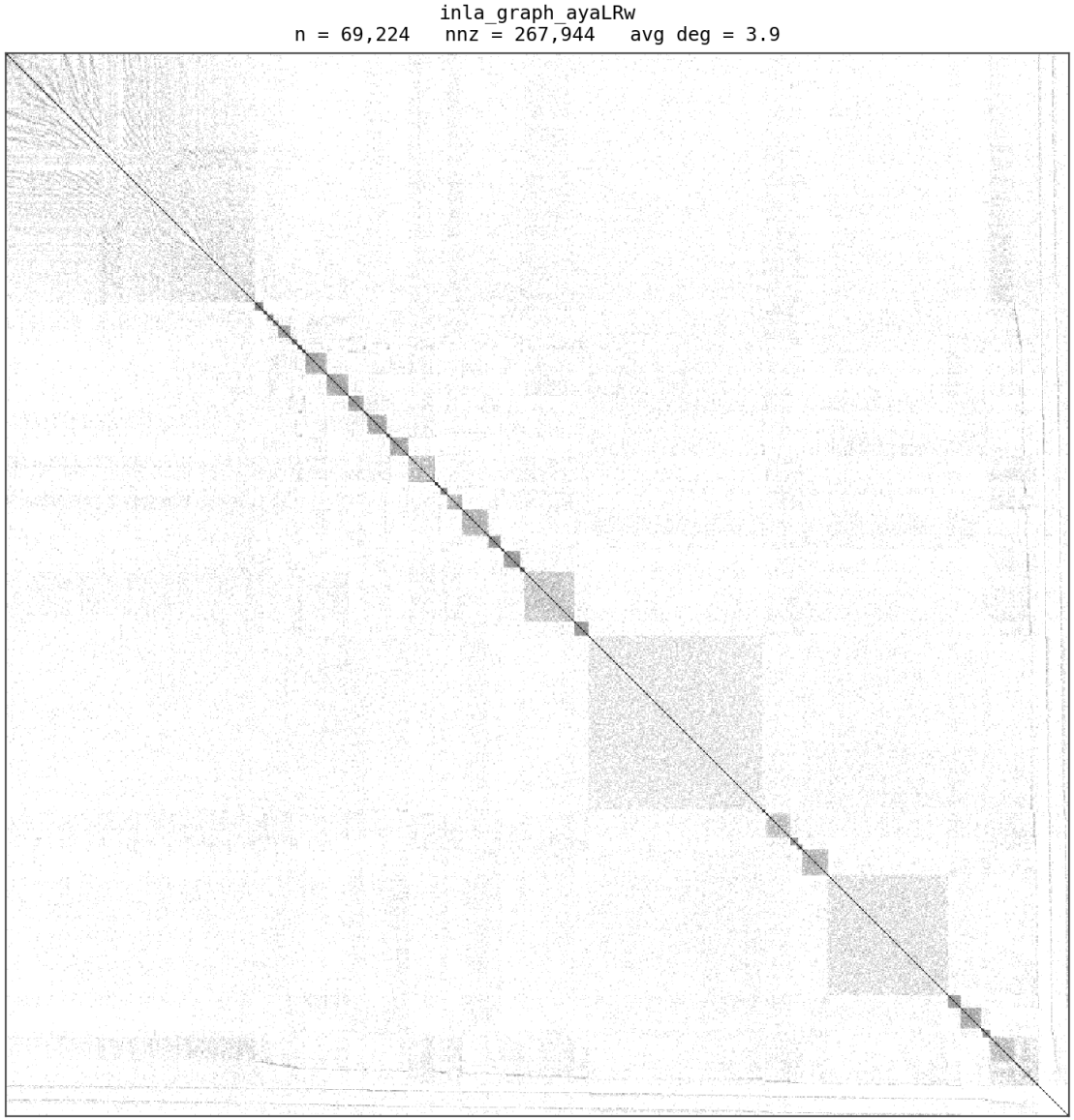
- Hyperparameters $\boldsymbol{\theta}$ (latent and/or likelihood)
- Each component is a GMRF (or a "small" dense)
- Covariance is a property, not a defining quantity



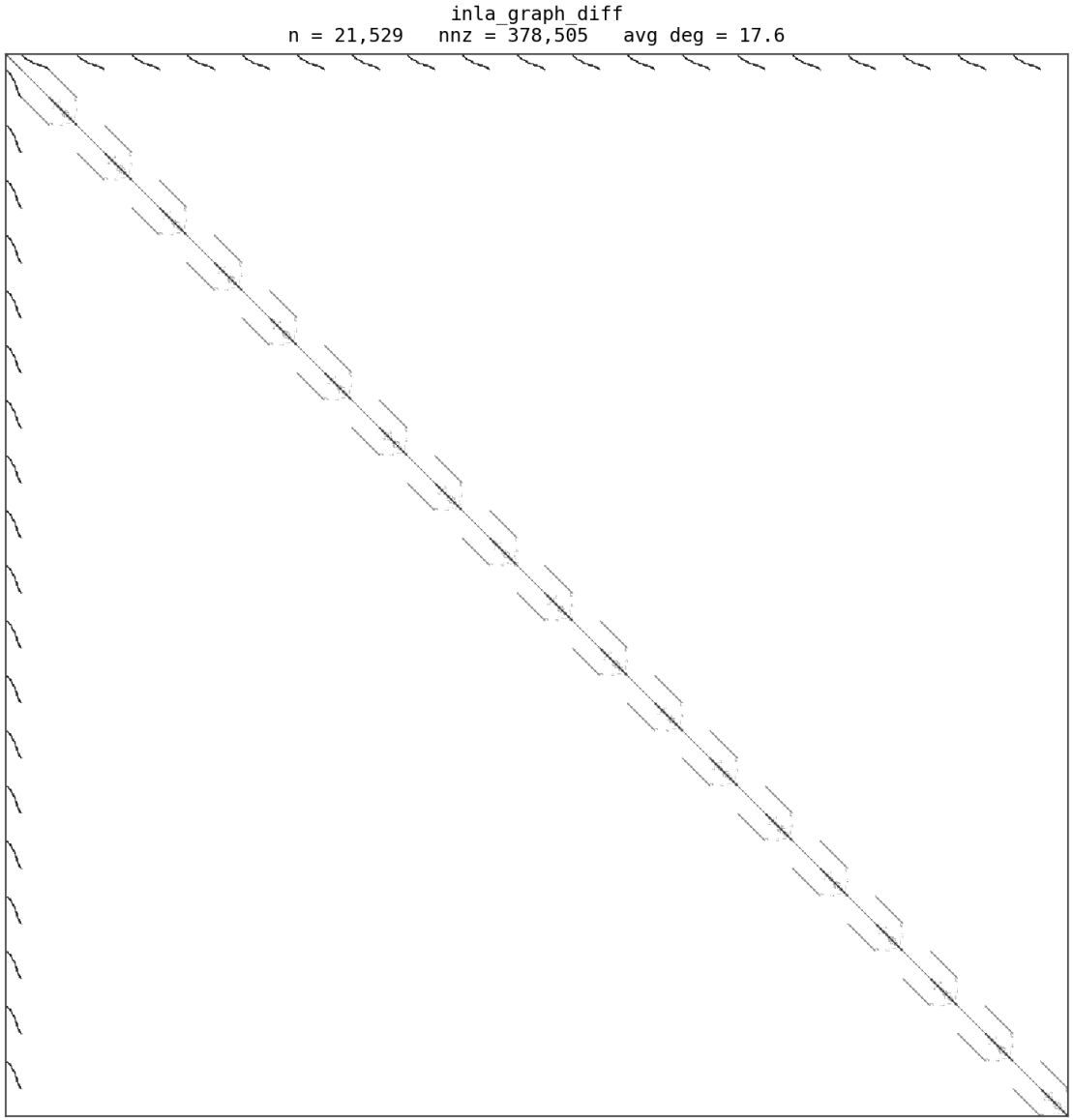
Examples



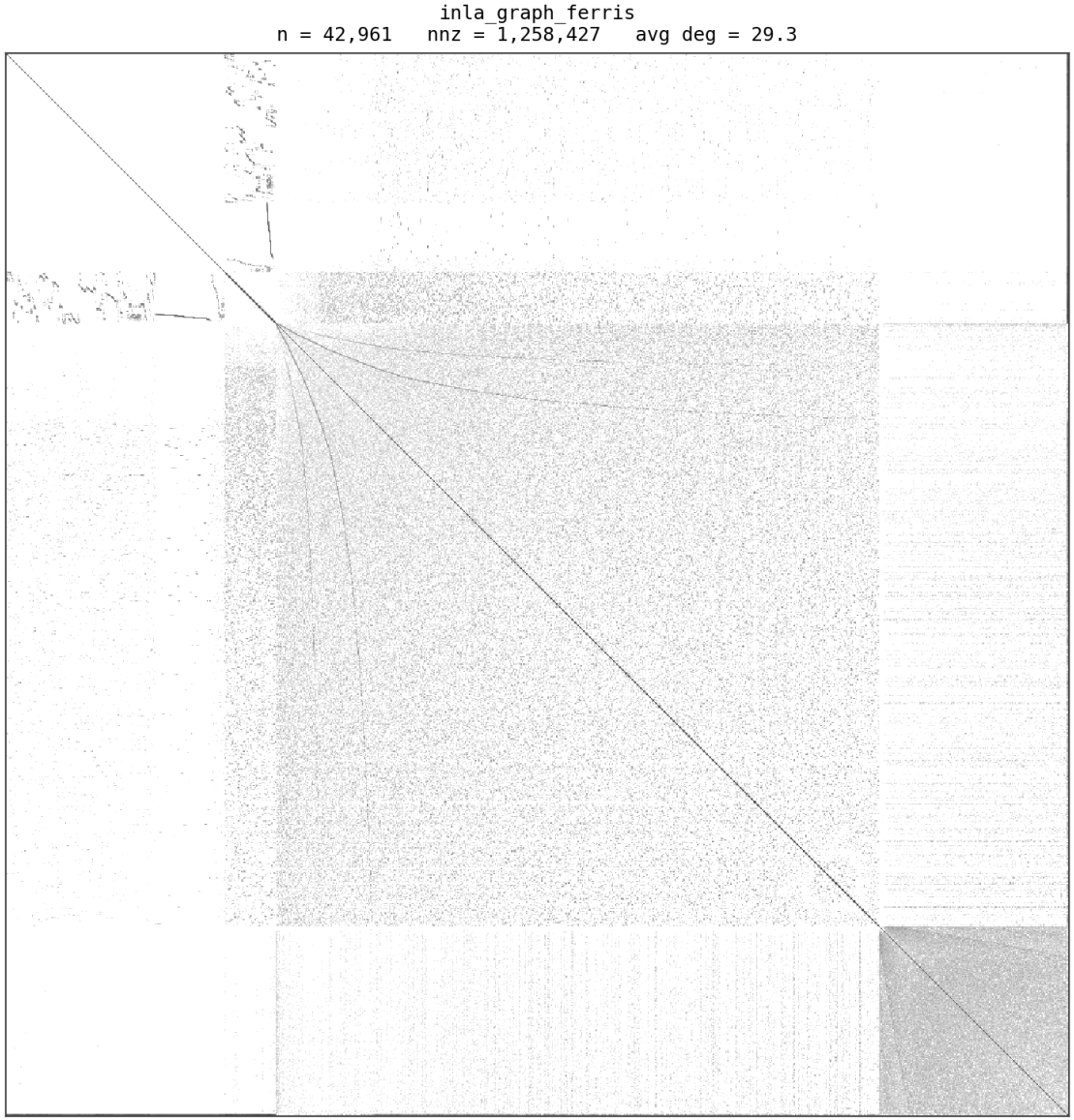
Examples



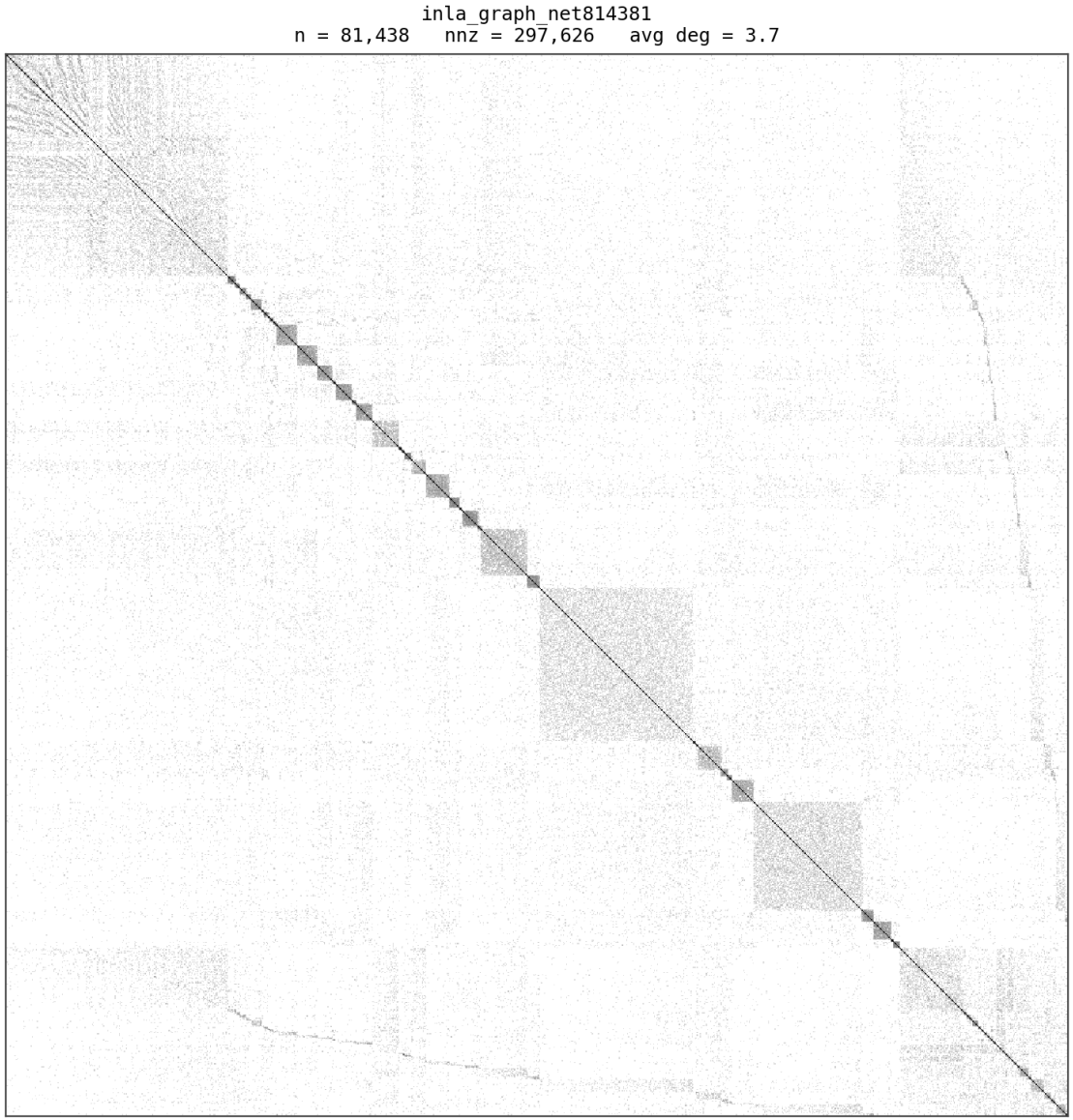
Examples



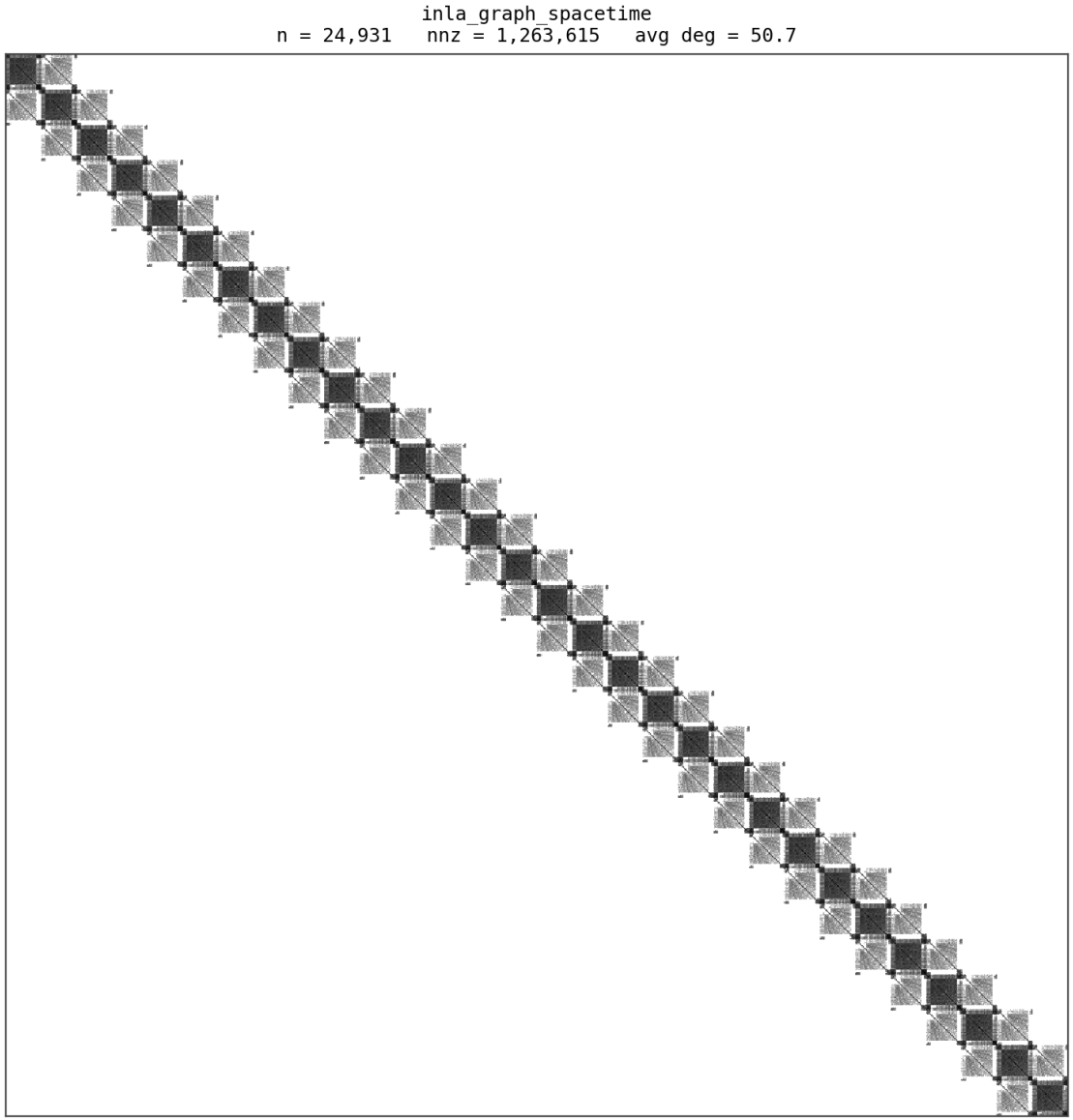
Examples



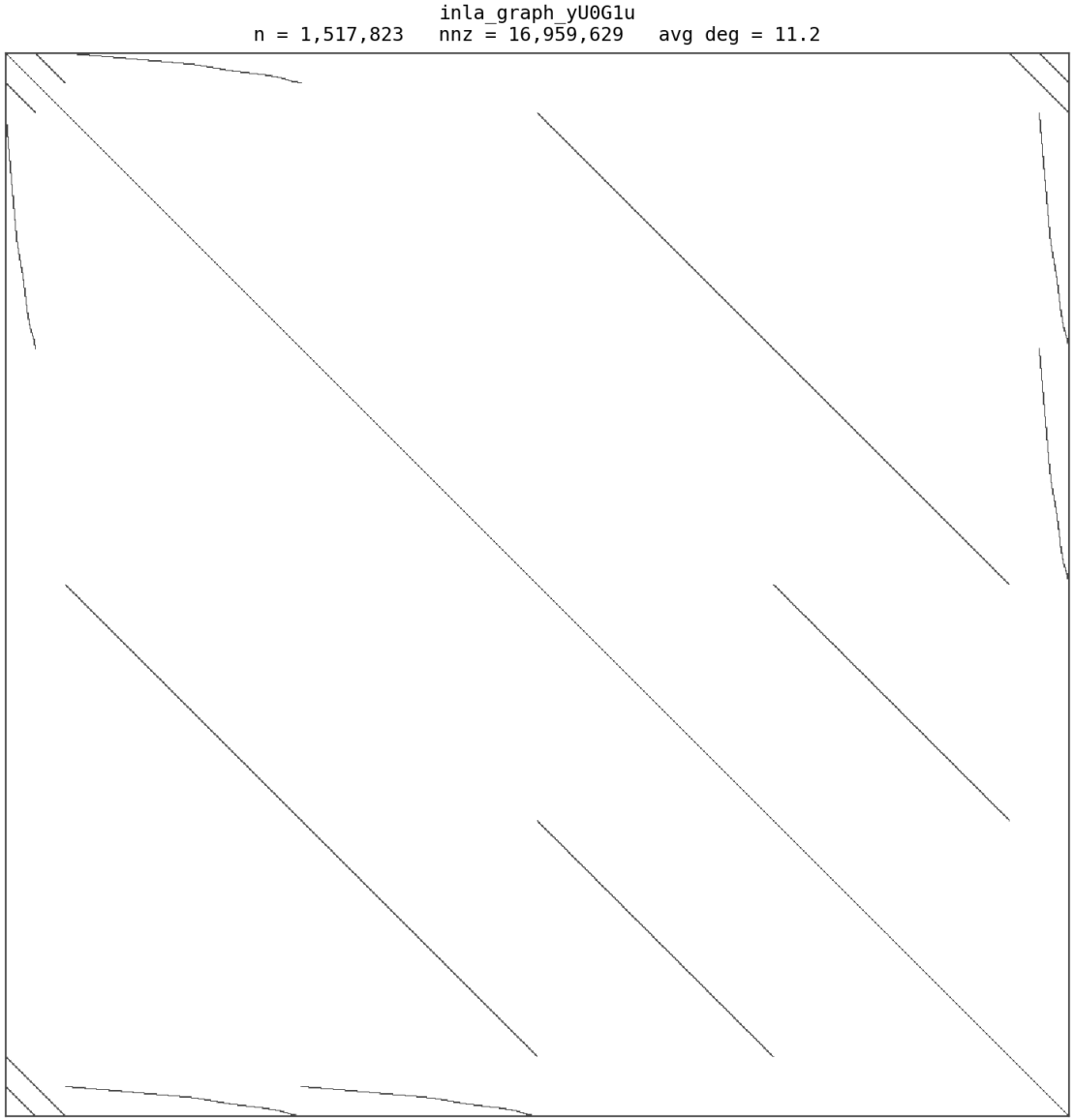
Examples



Examples



Examples



Bayesian Inference: simple ideas

- Gaussian approximations
- Laplace approximations
- Variational Bayes corrections
- Numerical integration
- Available in `R-INLA` package (`www.r-inla.org`)

Bayesian Inference: simple ideas

- Gaussian approximations
- Laplace approximations
- Variational Bayes corrections
- Numerical integration
- Available in `R-INLA` package (`www.r-inla.org`)

Bayesian Inference: simple ideas

- Gaussian approximations
- Laplace approximations
- Variational Bayes corrections
- Numerical integration
- Available in `R-INLA` package (`www.r-inla.org`)

Bayesian Inference: simple ideas

- Gaussian approximations
- Laplace approximations
- Variational Bayes corrections
- Numerical integration
- Available in `R-INLA` package (`www.r-inla.org`)

Bayesian Inference: simple ideas

- Gaussian approximations
- Laplace approximations
- Variational Bayes corrections
- Numerical integration
- Available in `R-INLA` package (`www.r-inla.org`)

Computational issues

Numerical methods for sparse matrices

- Factorise $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$
- Partial inverse $\boldsymbol{Q}^{-1}$ for all $i \sim j$ or $i = j$.
- Determinant $\log |\boldsymbol{Q}|$
- Solve $\boldsymbol{Q}\boldsymbol{x} = \boldsymbol{b}$ and $\boldsymbol{Q}\boldsymbol{X} = \boldsymbol{B}$

```
for (i = 0; i < dimensionSize; i++) {
    for (j = 0; j <= i; j++) {
        float sum = 0;
        for (k = 0; k < j; k++)
            sum += L[i][k] * L[j][k];

        if (i == j)
            L[i][j] = sqrt(A[i][i] - sum);
        else
            L[i][j] = (1.0 / L[j][j] * (A[i][j] - sum));
    }
}
```

Most important ingredient to scalable inference, is to handle sparse matrices well!

Computational issues

Numerical methods for sparse matrices

- Factorise $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$
- Partial inverse $\boldsymbol{Q}^{-1}$ for all $i \sim j$ or $i = j$.
- Determinant log $|\boldsymbol{Q}|$
- Solve $\boldsymbol{Q}\boldsymbol{x} = \boldsymbol{b}$ and $\boldsymbol{Q}\boldsymbol{X} = \boldsymbol{B}$

```
for (i = 0; i < dimensionSize; i++) {
    for (j = 0; j <= i; j++) {
        float sum = 0;
        for (k = 0; k < j; k++)
            sum += L[i][k] * L[j][k];

        if (i == j)
            L[i][j] = sqrt(A[i][i] - sum);
        else
            L[i][j] = (1.0 / L[j][j] * (A[i][j] - sum));
    }
}
```

Most important ingredient to scalable inference, is to handle sparse matrices well!

Computational issues

Numerical methods for sparse matrices

- Factorise $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$
- Partial inverse $\boldsymbol{Q}^{-1}$ for all $i \sim j$ or $i = j$.
- Determinant $\log |\boldsymbol{Q}|$
- Solve $\boldsymbol{Q}\boldsymbol{x} = \boldsymbol{b}$ and $\boldsymbol{Q}\boldsymbol{X} = \boldsymbol{B}$

```
for (i = 0; i < dimensionSize; i++) {
    for (j = 0; j <= i; j++) {
        float sum = 0;
        for (k = 0; k < j; k++)
            sum += L[i][k] * L[j][k];

        if (i == j)
            L[i][j] = sqrt(A[i][i] - sum);
        else
            L[i][j] = (1.0 / L[j][j] * (A[i][j] - sum));
    }
}
```

Most important ingredient to scalable inference, is to handle sparse matrices well!

Computational issues

Numerical methods for sparse matrices

- Factorise $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$
- Partial inverse $\boldsymbol{Q}^{-1}$ for all $i \sim j$ or $i = j$.
- Determinant $\log|\boldsymbol{Q}|$
- Solve $\boldsymbol{Q}\boldsymbol{x} = \boldsymbol{b}$ and $\boldsymbol{Q}\boldsymbol{X} = \boldsymbol{B}$

```
for (i = 0; i < dimensionSize; i++) {
    for (j = 0; j <= i; j++) {
        float sum = 0;
        for (k = 0; k < j; k++)
            sum += L[i][k] * L[j][k];

        if (i == j)
            L[i][j] = sqrt(A[i][i] - sum);
        else
            L[i][j] = (1.0 / L[j][j] * (A[i][j] - sum));
    }
}
```

Most important ingredient to scalable inference, is to handle sparse matrices well!

Computational issues

Numerical methods for sparse matrices

- Factorise $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$
- Partial inverse $\boldsymbol{Q}^{-1}$ for all $i \sim j$ or $i = j$.
- Determinant $\log|\boldsymbol{Q}|$
- Solve $\boldsymbol{Q}\boldsymbol{x} = \boldsymbol{b}$ and $\boldsymbol{Q}\boldsymbol{X} = \boldsymbol{B}$

```
for (i = 0; i < dimensionSize; i++) {
    for (j = 0; j <= i; j++) {
        float sum = 0;
        for (k = 0; k < j; k++)
            sum += L[i][k] * L[j][k];

        if (i == j)
            L[i][j] = sqrt(A[i][i] - sum);
        else
            L[i][j] = (1.0 / L[j][j] * (A[i][j] - sum));
    }
}
```

Most important ingredient to scalable inference, is to handle sparse matrices well!

Computational issues

Numerical methods for sparse matrices

- Factorise $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$
- Partial inverse $\boldsymbol{Q}^{-1}$ for all $i \sim j$ or $i = j$.
- Determinant $\log |\boldsymbol{Q}|$
- Solve $\boldsymbol{Q}\boldsymbol{x} = \boldsymbol{b}$ and $\boldsymbol{Q}\boldsymbol{X} = \boldsymbol{B}$

```
for (i = 0; i < dimensionSize; i++) {
    for (j = 0; j <= i; j++) {
        float sum = 0;
        for (k = 0; k < j; k++)
            sum += L[i][k] * L[j][k];

        if (i == j)
            L[i][j] = sqrt(A[i][i] - sum);
        else
            L[i][j] = (1.0 / L[j][j] * (A[i][j] - sum));
    }
}
```

Most important ingredient to scalable inference, is to handle sparse matrices well!

Not that much around...

- HPC community like “dense” more than “sparse”
- **R-INLA** uses TAUCS (outdated $>$ 20 years ago)
- CHOLMOD (what is used in R, MATLAB, does not scale well in parallel and do not handle inverse)
- PARDISO (good, not open-source and is now commercial, so...)
- etc...
- With the right person around, we can “just write our own”, right?

Esmail Abdul Fattah

Not that much around...

- HPC community like “dense” more than “sparse”
- **R-INLA** uses TAUCS (outdated $>$ 20 years ago)
- CHOLMOD (what is used in R, MATLAB, does not scale well in parallel and do not handle inverse)
- PARDISO (good, not open-source and is now commercial, so...)
- etc...
- With the right person around, we can “just write our own”, right?

Esmail Abdul Fattah

Not that much around...

- HPC community like “dense” more than “sparse”
- **R-INLA** uses TAUCS (outdated $>$ 20 years ago)
- CHOLMOD (what is used in R, MATLAB, does not scale well in parallel and do not handle inverse)
- PARDISO (good, not open-source and is now commercial, so...)
- etc...
- With the right person around, we can “just write our own”, right?

Esmail Abdul Fattah

Not that much around...

- HPC community like “dense” more than “sparse”
- **R-INLA** uses TAUCS (outdated $>$ 20 years ago)
- CHOLMOD (what is used in R, MATLAB, does not scale well in parallel and do not handle inverse)
- PARDISO (good, not open-source and is now commercial, so...)
- etc...
- With the right person around, we can “just write our own”, right?

Esmail Abdul Fattah

Not that much around...

- HPC community like “dense” more than “sparse”
- **R-INLA** uses TAUCS (outdated $>$ 20 years ago)
- CHOLMOD (what is used in R, MATLAB, does not scale well in parallel and do not handle inverse)
- PARDISO (good, not open-source and is now commercial, so...)
- etc...
- With the right person around, we can “just write our own”, right?

Esmail Abdul Fattah

Not that much around...

- HPC community like “dense” more than “sparse”
- **R-INLA** uses TAUCS (outdated $>$ 20 years ago)
- CHOLMOD (what is used in R, MATLAB, does not scale well in parallel and do not handle inverse)
- PARDISO (good, not open-source and is now commercial, so...)
- etc...
- With the right person around, we can “just write our own”, right?

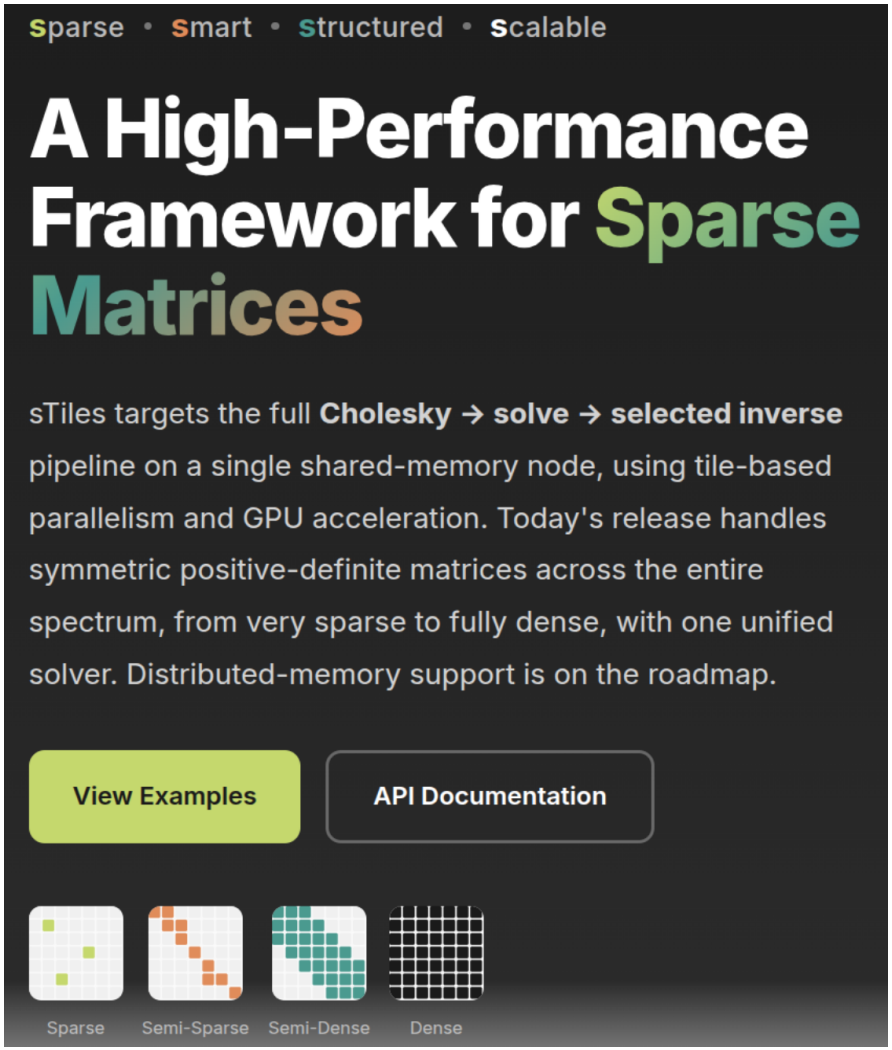
Esmail Abdul Fattah

Cholesky factorisation

```
for (i = 0; i < dimensionSize; i++) {
    for (j = 0; j <= i; j++) {
        float sum = 0;
        for (k = 0; k < j; k++)
            sum += L[i][k] * L[j][k];

        if (i == j)
            L[i][j] = sqrt(A[i][i] - sum);
        else
            L[i][j] = (1.0 / L[j][j] * (A[i][j] - sum));
    }
}
```

3 years later...



sTiles

3 years later...

- We're getting there
- Announced in Glasgow'25 in May
- ...the we found some new problematic cases
- that took another year to fix
- Written for modern CPU and will have GPU acceleration (for hard cases)
- Results are *very* good, with excellent parallel scaling
- Handle hard cases particulary well
- Open source when done
- Stay tuned!

sTiles

3 years later...

- We're getting there
- Announced in Glasgow'25 in May
- ...the we found some new problematic cases
- that took another year to fix
- Written for modern CPU and will have GPU acceleration (for hard cases)
- Results are *very* good, with excellent parallel scaling
- Handle hard cases particulary well
- Open source when done
- Stay tuned!

sTiles

3 years later...

- We're getting there
- Announced in Glasgow'25 in May
- ...the we found some new problematic cases
- that took another year to fix
- Written for modern CPU and will have GPU acceleration (for hard cases)
- Results are *very* good, with excellent parallel scaling
- Handle hard cases particulary well
- Open source when done
- Stay tuned!

sTiles

3 years later...

- We're getting there
- Announced in Glasgow'25 in May
- ...the we found some new problematic cases
- that took another year to fix
- Written for modern CPU and will have GPU acceleration (for hard cases)
- Results are *very* good, with excellent parallel scaling
- Handle hard cases particulary well
- Open source when done
- Stay tuned!

sTiles

3 years later...

- We're getting there
- Announced in Glasgow'25 in May
- ...the we found some new problematic cases
- that took another year to fix
- Written for modern CPU and will have GPU acceleration (for hard cases)
- Results are *very* good, with excellent parallel scaling
- Handle hard cases particulary well
- Open source when done
- Stay tuned!

3 years later...

- We're getting there
- Announced in Glasgow'25 in May
- ...the we found some new problematic cases
- that took another year to fix
- Written for modern CPU and will have GPU acceleration (for hard cases)
- Results are *very* good, with excellent parallel scaling
- Handle hard cases particulary well
- Open source when done
- Stay tuned!

3 years later...

- We're getting there
- Announced in Glasgow'25 in May
- ...the we found some new problematic cases
- that took another year to fix
- Written for modern CPU and will have GPU acceleration (for hard cases)
- Results are *very* good, with excellent parallel scaling
- Handle hard cases particulary well
- Open source when done
- Stay tuned!

sTiles

3 years later...

- We're getting there
- Announced in Glasgow'25 in May
- ...the we found some new problematic cases
- that took another year to fix
- Written for modern CPU and will have GPU acceleration (for hard cases)
- Results are *very* good, with excellent parallel scaling
- Handle hard cases particulary well
- Open source when done
- Stay tuned!

3 years later...

- We're getting there
- Announced in Glasgow'25 in May
- ...the we found some new problematic cases
- that took another year to fix
- Written for modern CPU and will have GPU acceleration (for hard cases)
- Results are *very* good, with excellent parallel scaling
- Handle hard cases particulary well
- Open source when done
- Stay tuned!

sTiles

Thank you · شكرا

جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology