Lecture 5

Modelling areal data

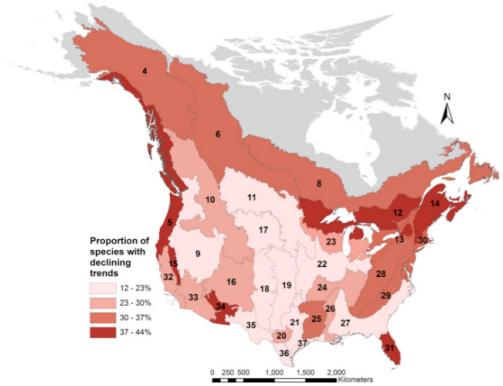
Sara Martino, Dept. of Mathematical Science, NTNU,
Janine Illian,
Jafet Belmont, University of Glasgow

March 7, 2026

Recall: Types of spatial data

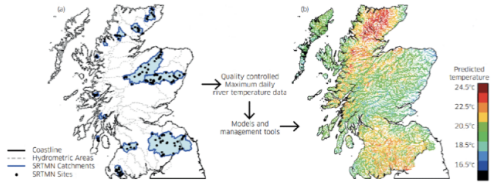
We can distinguish three types of spatial data structures

Areal data



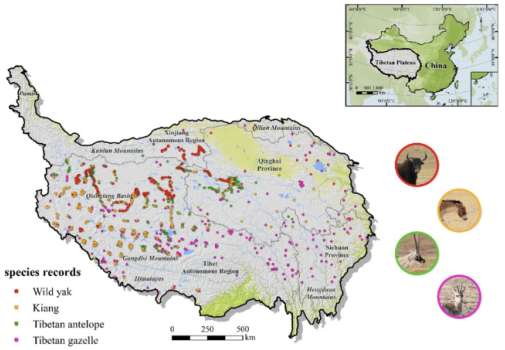
Map of bird conservation regions (BCRs) showing the proportion of bird species within each region showing a declining trend

Geostatistical data



Scotland river temperature monitoring network

Point-referenced data



Occurrence records of four ungulate species in the Tibet,

Types of spatial data

Recall:

Discrete space:

- **Data on a spatial grid (areal data)**

Continuous space:

- Geostatistical (geo-referenced) data
- Spatial point pattern data

Model components are used to reflect spatial dependence structures in discrete and continuous space.

areal data – model component

discrete data in space are related to discrete data in time

areal data – model component

discrete data in space are related to discrete data in time

$\rightsquigarrow$ very similar thinking

- neighbourhood structure – neighbours in space
- e.g. we can consider random walk models in space

areal data – model component

discrete data in space are related to discrete data in time

$\rightsquigarrow$ very similar thinking

- neighbourhood structure – neighbours in space
- e.g. we can consider random walk models in space
- these are special cases of **Gauss Markov random fields** (GMRF)
- recall: latent **Gaussian** models...

large latent Gaussian models

- real-world applications often involve latent fields with $n = 10^2$–10^5 elements
- computation times and memory issues become a problem...

large latent Gaussian models

- real-world applications often involve latent fields with $n = 10^2$–10^5 elements
- computation times and memory issues become a problem...

$\rightsquigarrow$ we need simplifications to make these feasible

- GMRFs provide a computationally efficient representation

large latent Gaussian models

- real-world applications often involve latent fields with $n = 10^2$–10^5 elements
- computation times and memory issues become a problem...

$\rightsquigarrow$ we need simplifications to make these feasible

- GMRFs provide a computationally efficient representation
- we will see later that these will come in handy even for continuous data (the "SPDE approach")

The Gaussian distribution

A Gaussian distribution $x \sim \mathcal{N}(0, \Sigma)$ is controlled by:

- The mean vector, which is set equal to 0 here, is the centre of the distribution

The Gaussian distribution

A Gaussian distribution $x \sim \mathcal{N}(0, \Sigma)$ is controlled by:

- The mean vector, which is set equal to 0 here, is the centre of the distribution
- The matrix Σ describes all pairwise covariances $\Sigma_{i,j} = \mathrm{Cov}[x_i, x_j]$
- Analytically tractable and well understood...

Formally

$$\pi(x) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}x^T\Sigma^{-1}x}$$

Computational Issues

- Σ is typically large and dense
- infeasible for large n
- we need to reduce computational cost

Sparse Matrices

A matrix Q is **sparse** if most of its elements are zero.

- Efficient numerical algorithms exist for sparse systems.
- We can exploit sparsity to make computations feasible.

Sparse Matrices

A matrix Q is **sparse** if most of its elements are zero.

- Efficient numerical algorithms exist for sparse systems.
- We can exploit sparsity to make computations feasible.

Two Options

1. Force Σ to be sparse $\Rightarrow$ independence...

Sparse Matrices

A matrix Q is **sparse** if most of its elements are zero.

- Efficient numerical algorithms exist for sparse systems.
- We can exploit sparsity to make computations feasible.

Two Options

1. Force Σ to be sparse $\Rightarrow$ independence...
2. Force $Q = \Sigma^{-1}$ (the precision) to be sparse $\Rightarrow$ conditional independence

Conditional independence

Example: AR(1) Process

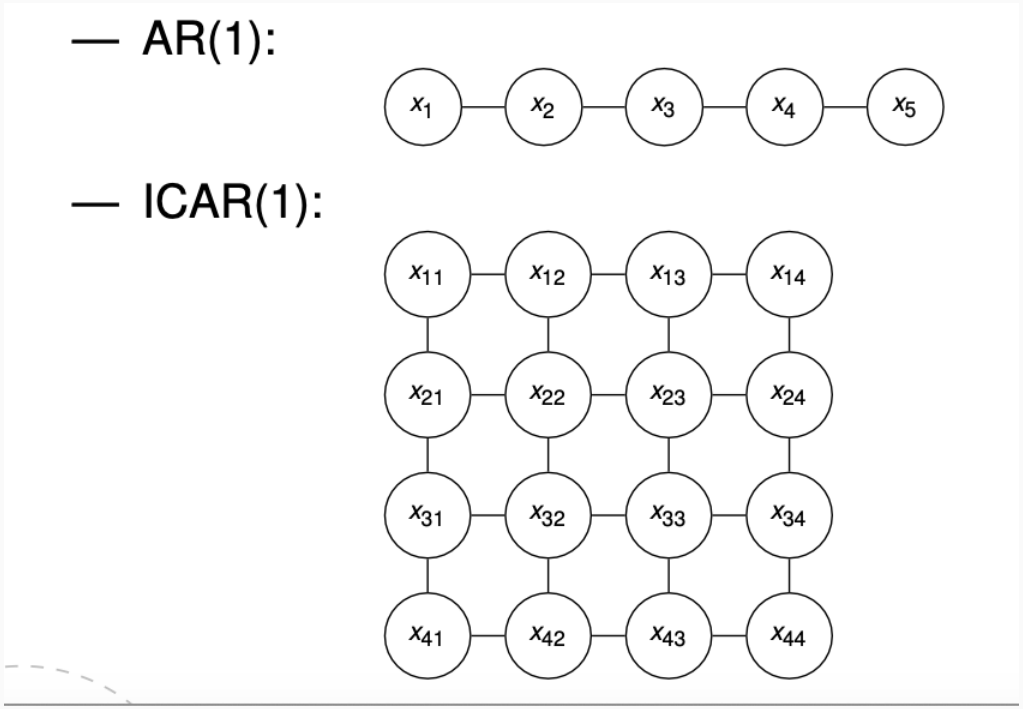
$$x_1 \sim \mathcal{N}\left(0, \frac{1}{1-\phi^2}\right),$$

$x_t = \phi x_{t-1} + \varepsilon_t,\ \varepsilon_t \sim \mathcal{N}(0,1)$

- Σ is dense.
- But Σ^{-1} is tridiagonal (sparse).

Graphical Representation

- Conditional independence structure can be represented via a graph.
- Each node corresponds to a variable x_i.
- Edges correspond to nonzero Q_{ij}.



Definition of a GMRF

A random vector x is a **Gaussian Markov Random Field (GMRF)** on a graph $G = (V, E)$ with mean μ and precision Q if:

$$\pi(x) = \frac{|Q|^{1/2}}{(2\pi)^{n/2}} \exp\left[-\frac{1}{2}(x-\mu)^T Q(x-\mu)\right],$$

and $Q_{ij} \neq 0 \Leftrightarrow \{i, j\} \in E$.

Example: AR(1) process

$$x_1 \sim \mathcal{N}\left(0, \frac{1}{1-\phi^2}\right)$$

$$x_t = \phi x_{t-1} + \varepsilon_t, \quad t > 1, \quad \varepsilon_t \sim \mathcal{N}(0,1)$$

The covariance matrix is dense since all time points are dependent, but the precision matrix Σ^1 is not:

$$\Sigma^{-1} = \begin{bmatrix} 1 & -\phi & 0 & \cdots & 0 \\ -\phi & 1+\phi^2 & -\phi & \cdots & 0 \\ 0 & -\phi & 1+\phi^2 & -\phi & \vdots \\ \vdots & \vdots & -\phi & 1+\phi^2 & -\phi \\ 0 & \cdots & 0 & -\phi & 1 \end{bmatrix}$$

Example: AR(1) process

- the tridiagonal form of Σ^{-1} can be exploited for quick calculations
- we call $Q = \Sigma^{-1}$ the precision matrix.
- key property that causes Σ^{-1} to be sparse – conditional independence

Example: AR(1) process

- the tridiagonal form of Σ^{-1} can be exploited for quick calculations
- we call $Q = \Sigma^{-1}$ the precision matrix.
- key property that causes Σ^{-1} to be sparse – conditional independence
- Can we take advantage of it in other models?

(Informal) definition of a GMRF

- a GMRF is a Gaussian distribution where the non-zero elements of the precision matrix are defined by a graph structure

(Informal) definition of a GMRF

- a GMRF is a Gaussian distribution where the non-zero elements of the precision matrix are defined by a graph structure
- in the time series AR1 example each variable is connected only to its predecessor and successor

$\rightsquigarrow$ the precision matrix is tridiagonal, i.e. sparse

(Informal) definition of a GMRF

- a GMRF is a Gaussian distribution where the non-zero elements of the precision matrix are defined by a graph structure
- in the time series AR1 example each variable is connected only to its predecessor and successor

$\rightsquigarrow$ the precision matrix is tridiagonal, i.e. sparse

$\rightsquigarrow$ efficient computation

example of a spatial GMRF

iCAR or Besag model:

- each region conditionally has a Gaussian distribution with
  - mean equal to the average of the neighbours and
  - precision proportional to the number of neighbours

example of a spatial GMRF

iCAR or Besag model:

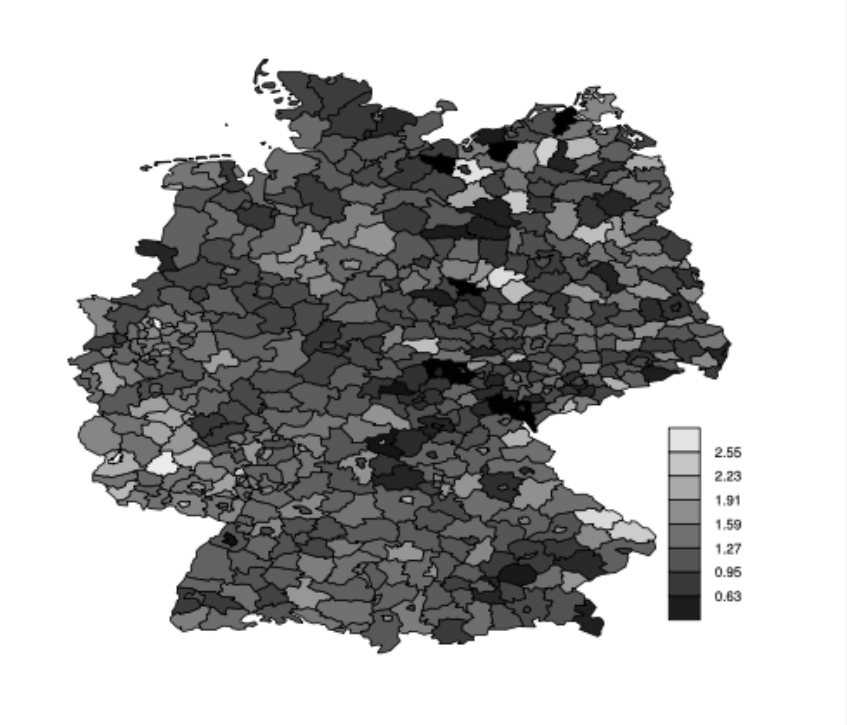
- each region conditionally has a Gaussian distribution with
  - mean equal to the average of the neighbours and
  - precision proportional to the number of neighbours
- also know as RW1 (in 2D) model

example of a spatial GMRF

iCAR or Besag model:

- each region conditionally has a Gaussian distribution with
  - mean equal to the average of the neighbours and
  - precision proportional to the number of neighbours
- also know as RW1 (in 2D) model

example: larynx cancer data from Germany

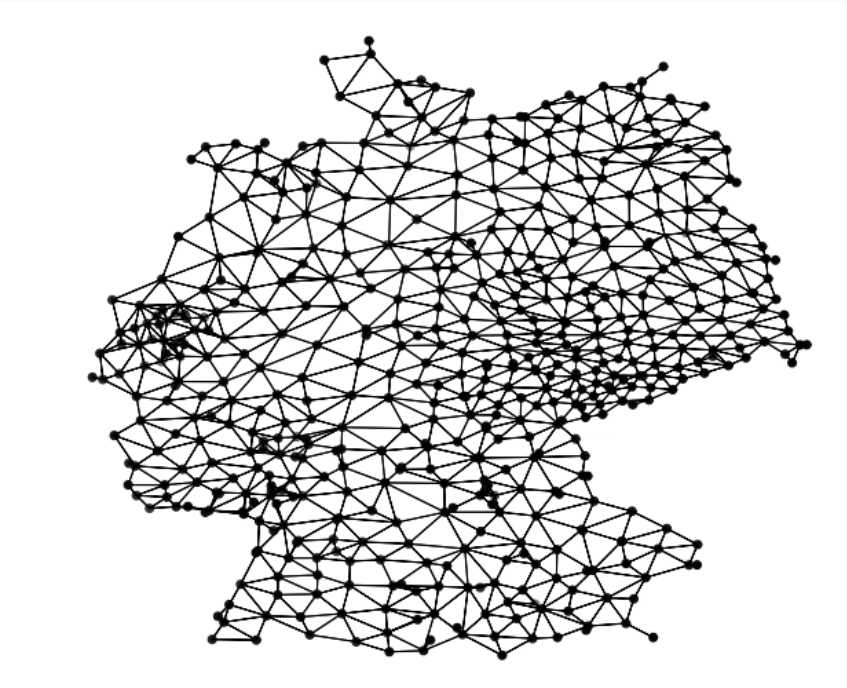


example of a spatial GMRF

iCAR or Besag model:

- each region conditionally has a Gaussian distribution with
  - mean equal to the average of the neighbours and
  - precision proportional to the number of neighbours
- also know as RW1 (in 2D) model

example: larynx cancer data from Germany

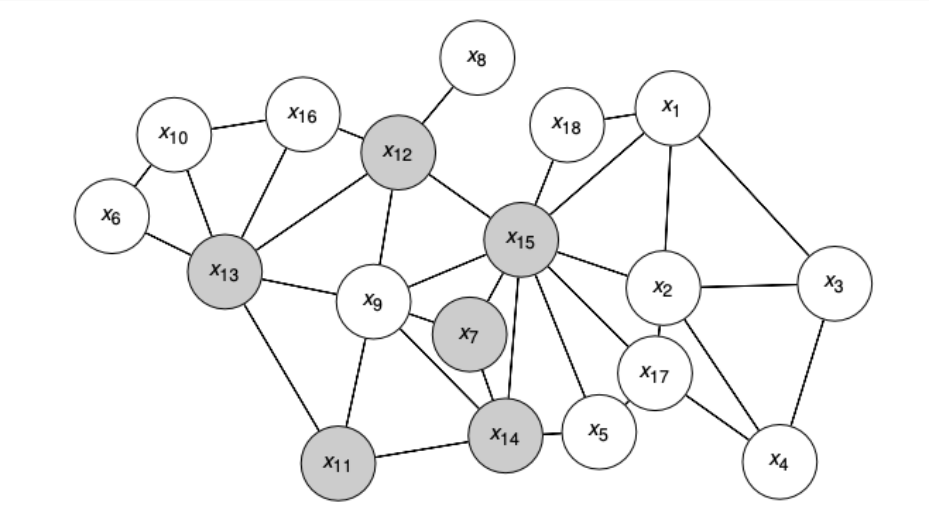


example of a spatial GMRF

iCAR or Besag model:

- each region conditionally has a Gaussian distribution with
  - mean equal to the average of the neighbours and
  - precision proportional to the number of neighbours
- also know as RW1 (in 2D) model

example: larynx cancer data from Germany

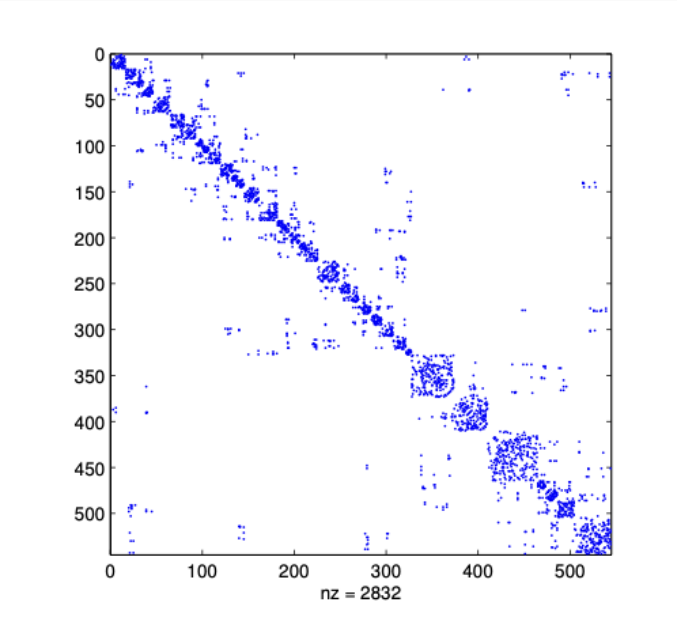


example of a spatial GMRF

iCAR or Besag model:

- each region conditionally has a Gaussian distribution with
  - mean equal to the average of the neighbours and
  - precision proportional to the number of neighbours
- also know as RW1 (in 2D)

example: larynx cancer data from Germany – precision matrix of the sub graph



And now...?

What do we need the GMRF models for?

And now...?

What do we need the GMRF models for?

recall: "Model components are used to reflect spatial dependence structures in discrete and continuous space."

And now...?

What do we need the GMRF models for?

recall: "Model components are used to reflect spatial dependence structures in discrete and continuous space."

spatial structures in a model are explained by

- covariate(s) – relationship between covariate and response might be of interest

And now...?

What do we need the GMRF models for?

recall: "Model components are used to reflect spatial dependence structures in discrete and continuous space."

spatial structures in a model are explained by

- covariate(s) – relationship between covariate and response might be of interest
- an GMRF model – to account for additional dependence for valid inference
- potentially some additional random error

And now...?

What do we need the GMRF models for?
recall: "Model components are used to reflect spatial dependence structures in discrete and continuous space."
spatial structures in a model are explained by

- covariate(s) – relationship between covariate and response might be of interest
- an GMRF model – to account for additional dependence for valid inference
- potentially some additional random error

Here – BYM (Besag-York-Mollié) model:

$$y_i|\eta_i \sim \text{Poisson}(E_i\lambda_i)$$

$$\eta_i = \log(\lambda_i) = \beta_0 + \beta_1 pm(10) + \omega_i + z_i$$

The BYM model in `inlabru`

The Model

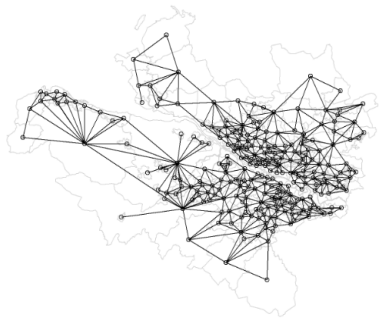
$$y_i|\eta_t \sim \text{Poisson}(E_i\lambda_i)$$
$$\log(\lambda_i) = \eta_i = \beta_0 + \beta_1 \text{pm10} + u_i + z_i$$

The code

```
# define model component
cmp = ~ Intercept(1) +  pm10(pm10, model = "linear") +
  space(space, model = "besag", graph = Q) +
  iid(space, model = "iid")

# define model predictor
eta =  observed ~ Intercept + space + iid

# build the observation model
lik = bru_obs(formula = formula,
              family = "poisson",
              E = expected,
              data = resp_cases)

# fit the model
fit = bru(cmp, lik)
```

neighbourhood structure

```
library(spdep)
W.nb <- poly2nb(GGHB.IZ,queen = TRUE)
```



The BYM model in `inlabru`

The Model

$$y_i | \eta_t \sim \text{Poisson}(E_i \lambda_i)$$

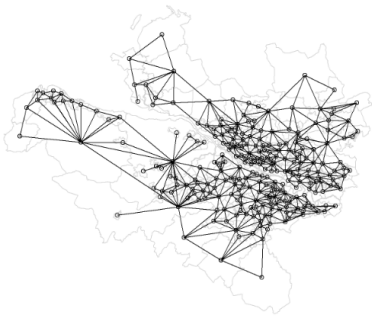
$$\log(\lambda_i) = \boxed{\eta_i} = \boxed{\beta_0 + \beta_1 \text{pm10} + u_i + z_i}$$

The code

```
# define model component
cmp = ~ Intercept(1) + pm10(pm10, model = "linear")
  space(space, model = "besag", graph = Q) +
  iid(space, model = "iid")

# define model predictor
eta =  observed ~ Intercept + space + iid

# build the observation model
lik = bru_obs(formula = formula,
              family = "poisson",
              E = expected,
              data = resp_cases)

# fit the model
fit = bru(cmp, lik)
```

neighbourhood structure

```
library(spdep)
W.nb <- poly2nb(GGHB.IZ,queen = TRUE)
```



The BYM model in `inlabru`

The Model

$$y_i|\eta_t \sim \text{Poisson}(E_i\lambda_i)$$

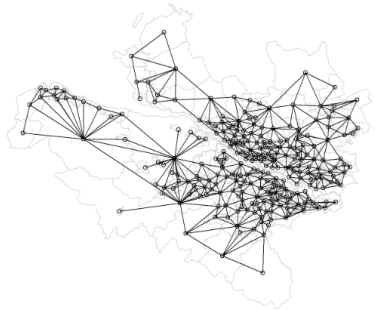
$$\log(\lambda_i) = \eta_i = \beta_0 + \beta_1 \text{pm10} + u_i + z_i$$

The code

```
# define model component
cmp = ~ Intercept(1) + pm10(pm10, model = "linear")
  space(space, model = "besag", graph = Q) +
  iid(space, model = "iid")

# define model predictor
eta =  observed ~ Intercept + space + iid

# build the observation model
lik = bru_obs(formula = formula,
              family = "poisson",
              E = expected,
              data = resp_cases)

# fit the model
fit = bru(cmp, lik)
```

neighbourhood structure

```
library(spdep)
W.nb <- poly2nb(GGHB.IZ,queen = TRUE)
```



The BYM model in `inlabru`

The Model

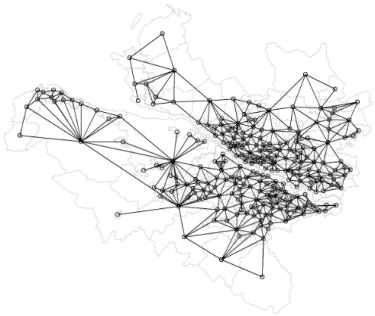
$$y_i|\eta_t \sim \text{Poisson}(E_i\lambda_i)$$
$$\log(\lambda_i) = \eta_i = \beta_0 + \beta_1 \text{pm10} + u_i + z_i$$

The code

```
# define model component
cmp = ~ Intercept(1) + pm10(pm10, model = "linear")
  space(space, model = "besag", graph = Q) +
  iid(space, model = "iid")

# define model predictor
eta =  observed ~ Intercept + space + iid

# build the observation model
lik = bru_obs(formula = formula,
              family = "poisson",
              E = expected,
              data = resp_cases)

# fit the model
fit = bru(cmp, lik)
```

neighbourhood structure

```
library(spdep)
W.nb <- poly2nb(GGHB.IZ,queen = TRUE)
```



Take-home message

- areal data are data on discretised space
- spatial relationships are represented through a neighbourhood structure
- GMRFs are a computationally efficient representation
- these will also be relevant – and very useful – when it comes to approximating spatial structures in continuous space

Take-home message

- areal data are data on discretised space
- spatial relationships are represented through a neighbourhood structure
- GMRFs are a computationally efficient representation
- these will also be relevant – and very useful – when it comes to approximating spatial structures in continuous space
- look forward to GRMFs in continuously indexed space...