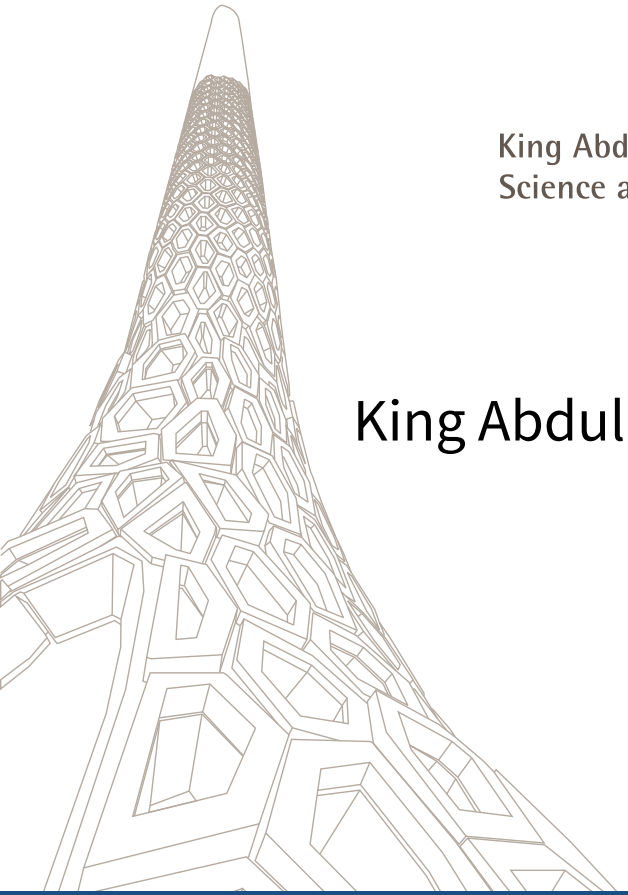
Bayesian computing with INLA



King Abdullah University of
Science and Technology

جامعة الملك عبدالله
للعلوم والتقنية

Håvard Rue
King Abdullah University of Science and Technology
Saudi-Arabia

Oct 2018

What is INLA?

- INLA do approximate Bayesian inference for a spesific class of models
- This is the class of *latent Gaussian models*
- In most cases, INLA is faster and more accurate inference than with MCMC
- Cannot do *all models*, then the ones we can do, we do very very well.
- Many/most of the important models that people use, we can do

What is INLA?

- INLA do approximate Bayesian inference for a spesific class of models
- This is the class of *latent Gaussian models*
- In most cases, INLA is faster and more accurate inference than with MCMC
- Cannot do *all models*, then the ones we can do, we do very very well.
- Many/most of the important models that people use, we can do

What is INLA?

- INLA do approximate Bayesian inference for a spesific class of models
- This is the class of *latent Gaussian models*
- In most cases, INLA is faster and more accurate inference than with MCMC
- Cannot do *all models*, then the ones we can do, we do very very well.
- Many/most of the important models that people use, we can do

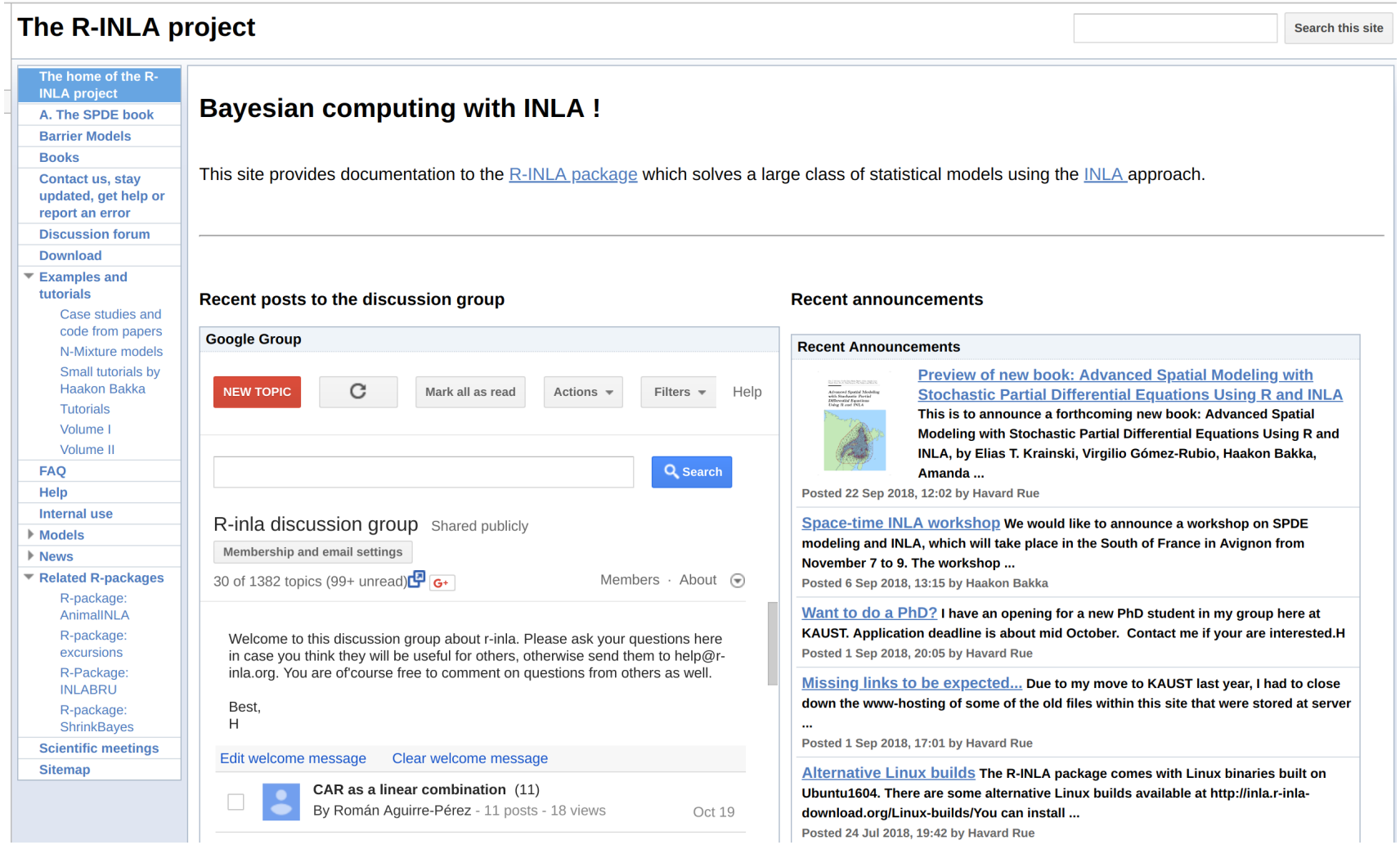
What is INLA?

- INLA do approximate Bayesian inference for a spesific class of models
- This is the class of *latent Gaussian models*
- In most cases, INLA is faster and more accurate inference than with MCMC
- Cannot do *all models*, then the ones we can do, we do very very well.
- Many/most of the important models that people use, we can do

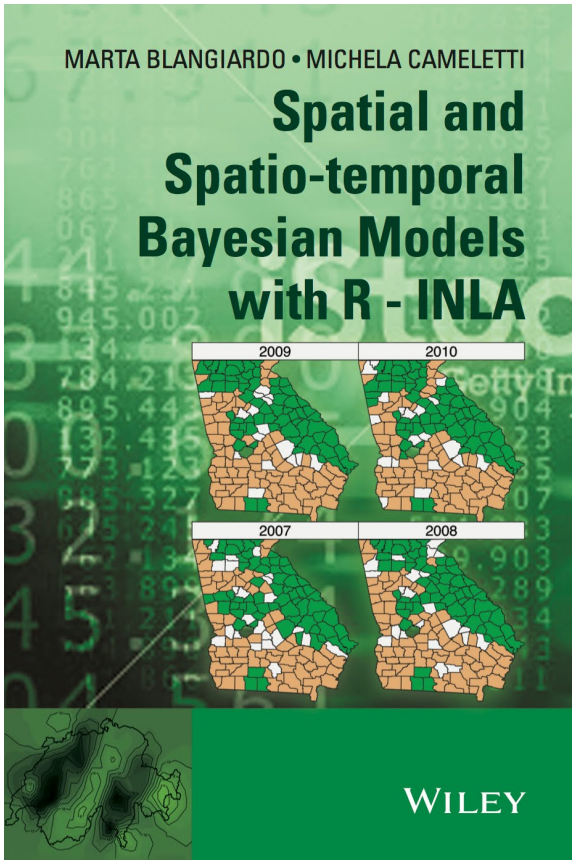
What is INLA?

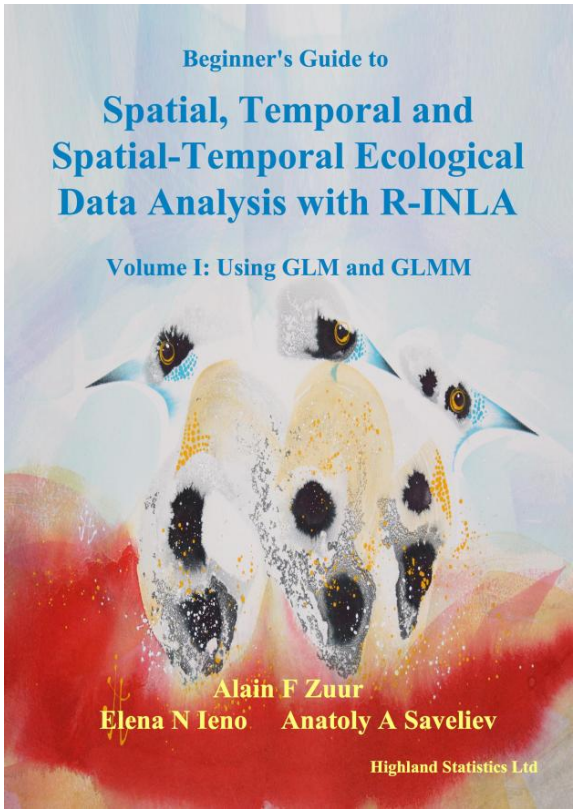
- INLA do approximate Bayesian inference for a spesific class of models
- This is the class of *latent Gaussian models*
- In most cases, INLA is faster and more accurate inference than with MCMC
- Cannot do *all models*, then the ones we can do, we do very very well.
- Many/most of the important models that people use, we can do

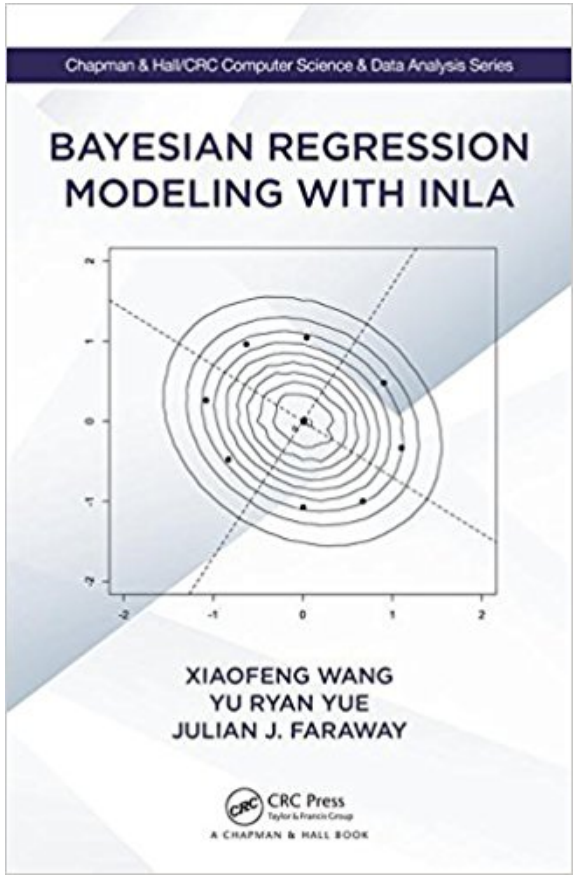
www.r-inla.org



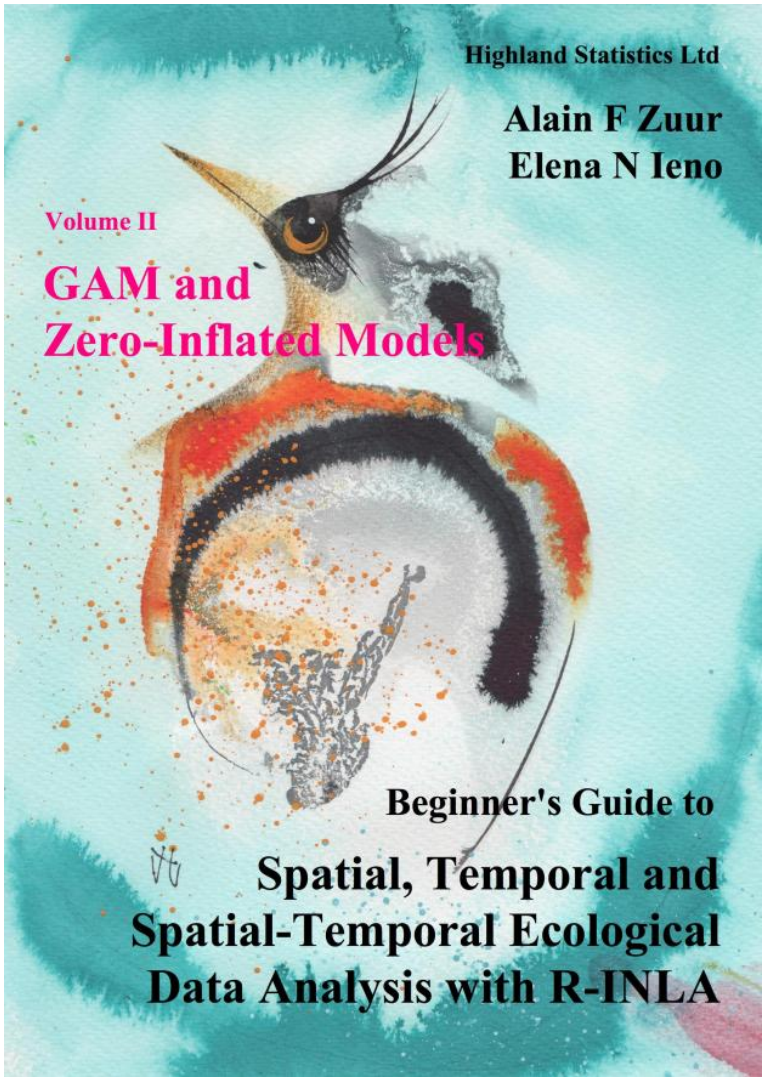
Books

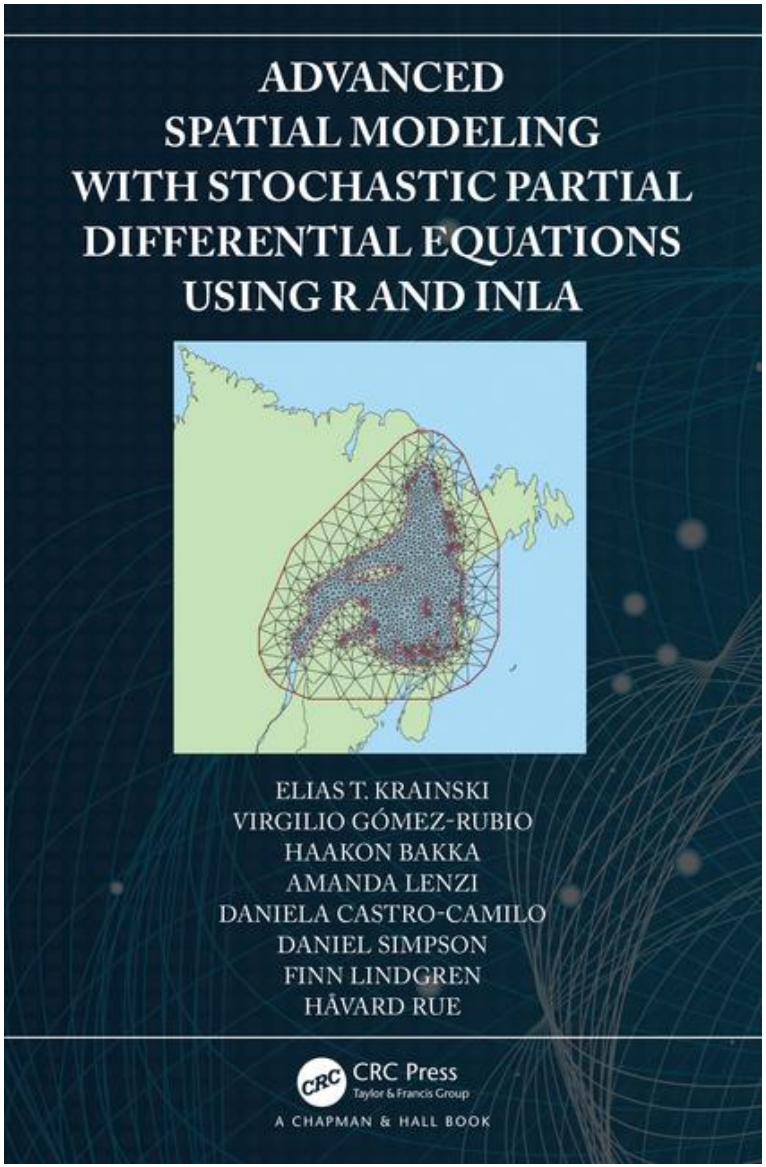

MARTA BLANGIARDO • MICHELA CAMELETTI
Spatial and Spatio-temporal Bayesian Models with R - INLA
2009
2010
2007
2008
WILEY


Beginner's Guide to
Spatial, Temporal and Spatial-Temporal Ecological Data Analysis with R-INLA
Volume I: Using GLM and GLMM
Alain F Zuur
Elena N Ieno
Anatoly A Saveliev
Highland Statistics Ltd


Chapman & Hall/CRC Computer Science & Data Analysis Series
BAYESIAN REGRESSION MODELING WITH INLA
XIAOFENG WANG
YU RYAN YUE
JULIAN J. FARAWAY
CRC Press
A CHAPMAN & HALL BOOK

Books, and more to come...





Some recent applications (this month)

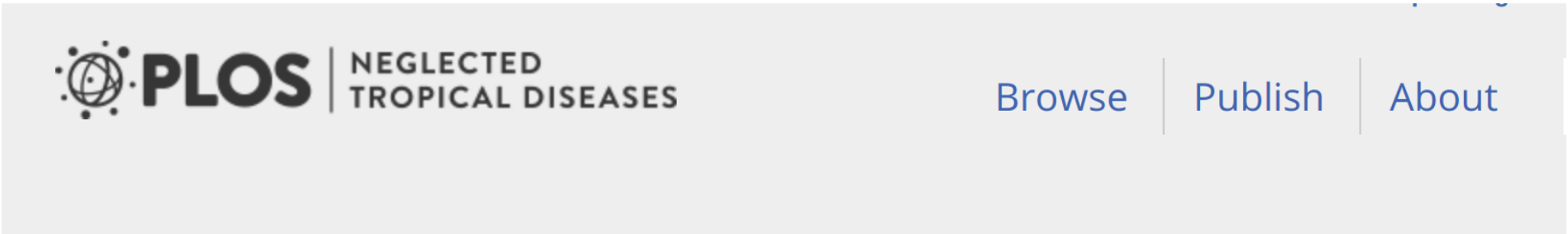
EFFICIENT RECONSTRUCTIONS OF COMMON ERA CLIMATE VIA INTEGRATED NESTED LAPLACE APPROXIMATIONS.

LUIS A. BARBOZA, JULIEN EMILE-GEAY, BO LI, AND WAN HE

ABSTRACT. A Paleoclimate Reconstruction on the Common Era (1-2000AD) was performed using a Hierarchical Bayesian Model from three types of data: proxy data from PAGES2k project dataset, HadCRUT4 temperature data from the Climatic Research Unit at the University of East Anglia, and external forcing data from several sources. Five data reduction techniques were explored with the purpose of achieving a parsimoneous but sufficient set of proxy equations. Instead of using the MCMC approach to solve for the latent variable, we employed an INLA algorithm that can approximate the MCMC results and meantime is much more computationally efficient than MCMC. The role of external forcings was investigated by replacing or combining them with a fixed number of BSplines in the latent equation. Two different validation exercises confirm that it is feasible to improve the predictive ability of traditional external forcing models.

1 [stat.AP] 14 Oct 2018

Some recent applications (this month)



OPEN ACCESS PEER-REVIEWED

RESEARCH ARTICLE

Chagas disease mortality in Brazil: A Bayesian analysis of age-period-cohort effects and forecasts for two decades

Taynãna César Simões, Laiane Félix Borges, Auzenda Conceição Parreira de Assis, Maria Vitórias Silva, Juliano dos Santos, Karina Cardoso Meira

Version 2 ▾ Published: September 28, 2018 • https://doi.org/10.1371/journal.pntd.0006798

Some recent applications (this month)

Home > Journals > Physics & Mathematics > OJS

Articles Archive Indexing Aims & Scope Editorial Board For Authors Publication Fees

OJS> Vol.8 No.5, October 2018 Share This Article: 1

Open Access

Spatio-Temporal Variation of HIV Infection in Kenya

Full-Text HTML XML Download as PDF (Size:1358KB) PP. 811-830

DOI: 10.4236/ojs.2018.85053 48 Downloads 114 Views

Author(s) Leave a comment

Benard Tonui[1*], Samuel Mwalili[2], Anthony Wanjoya[2]

Affiliation(s)

[1]Department of Mathematics & Computer Science, University of Kabianga, Kericho, Kenya.

[2]Department of Statistics & Actuarial Science, Jomo Kenyatta University of Agriculture & Technology, Juja, Kenya.

Some recent applications (this month)

Sociobiology

An international journal on social insects

HOME ABOUT LOGIN REGISTER CATEGORIES SEARCH CURRENT ARCHIVES
ANNOUNCEMENTS USER HELP INSTRUCTIONS FOR AUTHORS JOURNAL STATISTICS EDITORIAL BOARD

Home > Vol 65, No 3 (2018) > **Nielsen**

Experiences in Transplanting Wood Ants into Plantations for Integrated Pest Management

Jesper Stern Nielsen, Mogens Gissel Nielsen, Joachim Offenberg

Abstract

Ants can function efficiently as biocontrol agents in open field horticulture. Temperate wood ants can control forest pests, including species damaging forest regeneration plots and fruit plantations. Thus, they possess potential as biocontrol agents in open field horticulture, if they can persist in these systems. Here we present observationson activity and survival of wood ants transplanted from forests into different types of plantations. Mound fragments were transplanted into a conifer seedling plot, an organic and a conventional Christmas tree plantation, and

Some recent applications (this month)

Malaria Journal

Home About Articles Submission Guidelines

Methodology | Open Access

malariaAtlas: an R interface to global malariometric data hosted by the Malaria Atlas Project

Daniel A. Pfeffer, Timothy C. D. Lucas ✉, Daniel May, Joseph Harris, Jennifer Rozier, Katherine A. Twohig, Ursula Dalrymple, Carlos A. Guerra, Catherine L. Moyes, Mike Thorn, Michele Nguyen, Samir Bhatt, Ewan Cameron, Daniel J. Weiss, Rosalind E. Howes, Katherine E. Battle, Harry S. Gibson and Peter W. Gething

Malaria Journal 2018 **17**:352

https://doi.org/10.1186/s12936-018-2500-5

Received: 3 July 2018 | **Accepted:** 29 September 2018 | **Published:** 5 October 2018

Some recent applications (this month)

Access by
King Abdullah University of Science and Technology

JOURNALS

Research Article | Full Access

Larger plants promote a greater diversity of symbiotic nitrogen-fixing soil bacteria associated with an Australian endemic legume

Some recent applications (this month)

A three-stage model for short-term extreme wind speed probabilistic forecasting

Daniela Castro-Camilo[1], Raphaël Huser[1], and Håvard Rue[1]

October 8, 2018

Abstract

Renewable sources of energy such as wind power have become a sustainable alternative to fossil fuel-based energy. However, the uncertainty and fluctuation of the wind speed derived from its intermittent nature bring a great threat to the wind power production stability, and to the

9 Oct 2018

Some recent applications (this month)

nature > scientific reports > articles > article

SCIENTIFIC REPORTS

Article | OPEN | Published: 11 October 2018

Spatiotemporal dynamics and risk factors for human Leptospirosis in Brazil

Oswaldo Santos Baquero ✉ & Gustavo Machado ✉

Scientific Reports **8**, Article number: 15170 (2018) | Download Citation ↓

Some recent applications (this month)

RESEARCH ARTICLE | Full Access

A spatio-temporal approach to estimate patterns of climate change

M. P. Laurini

First published: 11 October 2018 | https://doi.org/10.1002/env.2542

Some recent applications (this month)

PHILOSOPHICAL TRANSACTIONS OF THE ROYAL SOCIETY B

BIOLOGICAL SCIENCES

search

Advanc

Home Content Information for About us Sign up Propose an issue

Check for updates

Interactive effects of tree size, crown exposure and logging on drought-induced mortality

Alexander Shenkin, Benjamin Bolker, Marielos Peña-Claros, Juan Carlos Licona, Nataly Ascarrunz, Francis E. Putz

Published 8 October 2018. DOI: 10.1098/rstb.2018.0189

Some recent applications (this month)

Coláiste na Tríonóide, Baile Átha Cliath
Trinity College Dublin
Ollscoil Átha Cliath | The University of Dublin

The Environmental Epidemiology of

Amyotrophic Lateral Sclerosis in Europe

A dissertation submitted to Trinity College Dublin in fulfilment for the award of

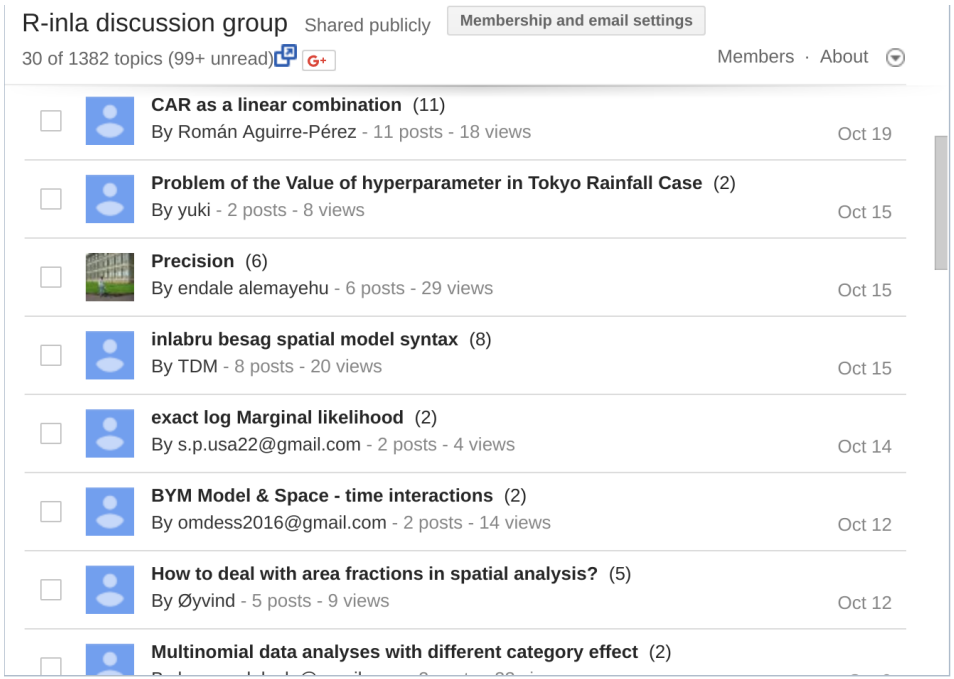
Doctor of Philosophy (PhD)

James PK Rooney

B.Sc. , M.B. B.Ch. B.A.O. , M.Sc.

Activities

Public discussion group



Activities

Public discussion group

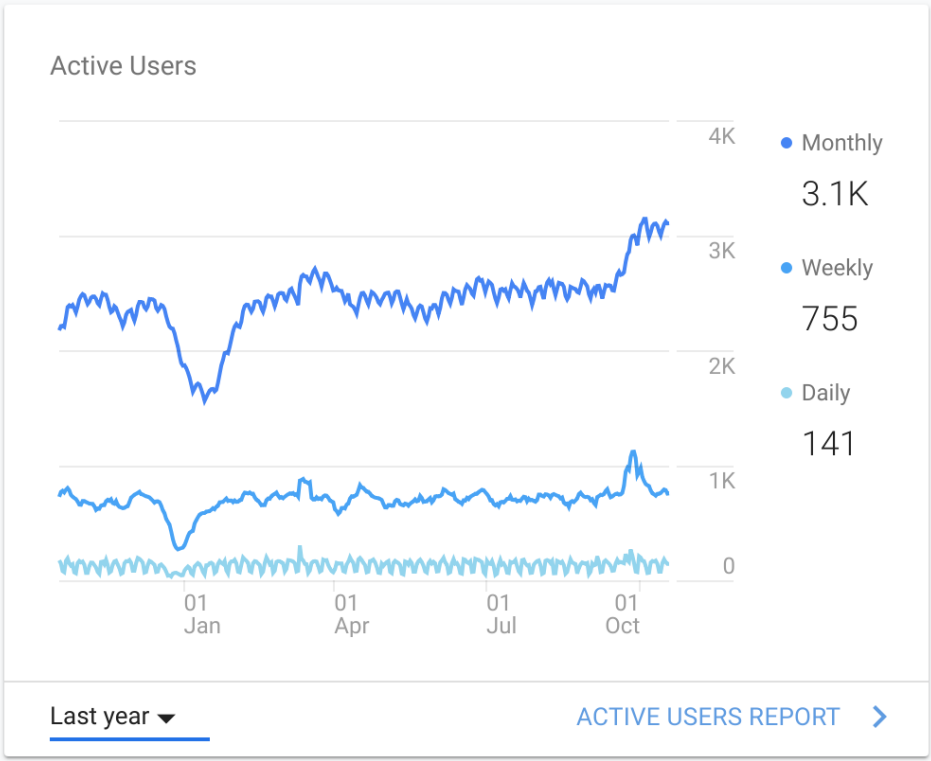
R-inla discussion group Shared publicly Membership and email settings

30 of 1382 topics (99+ unread) Members · About

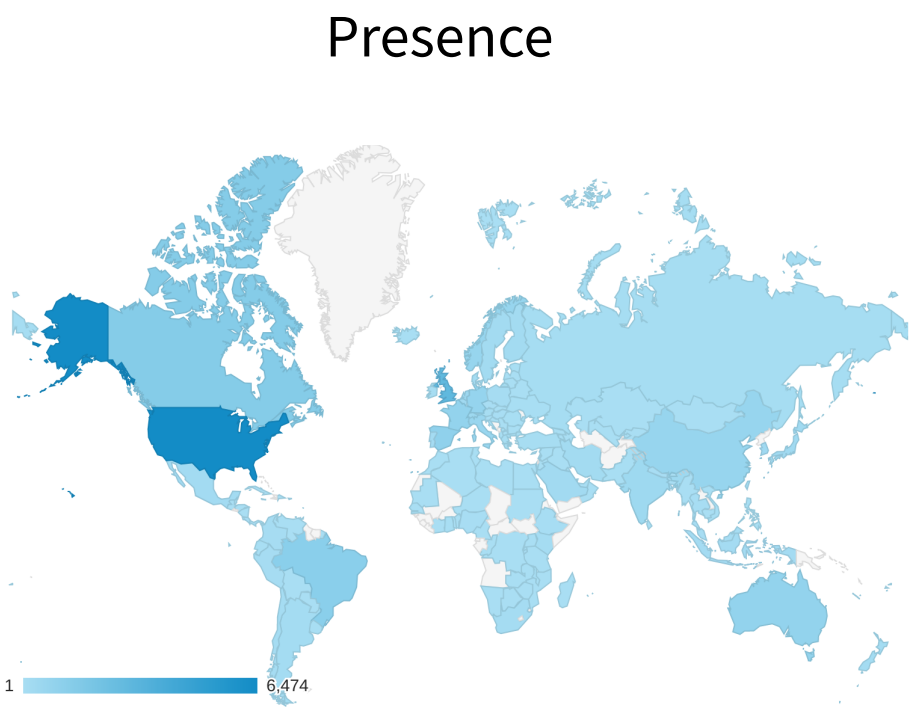
CAR as a linear combination (11) By Román Aguirre-Pérez - 11 posts - 18 views	Oct 19
Problem of the Value of hyperparameter in Tokyo Rainfall Case (2) By yuki - 2 posts - 8 views	Oct 15
Precision (6) By endale alemayehu - 6 posts - 29 views	Oct 15
inlabru besag spatial model syntax (8) By TDM - 8 posts - 20 views	Oct 15
exact log Marginal likelihood (2) By s.p.usa22@gmail.com - 2 posts - 4 views	Oct 14
BYM Model & Space - time interactions (2) By omdess2016@gmail.com - 2 posts - 14 views	Oct 12
How to deal with area fractions in spatial analysis? (5) By Øyvind - 5 posts - 9 views	Oct 12
Multinomial data analyses with different category effect (2)	

Visits to `www.r-inla.org`

How are your active users trending over time?

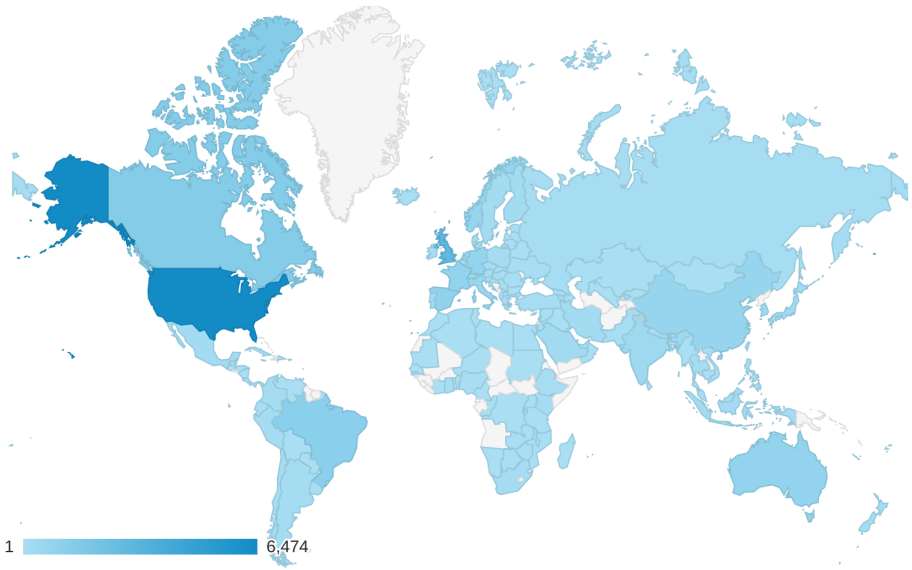


Locations of users



Locations of users

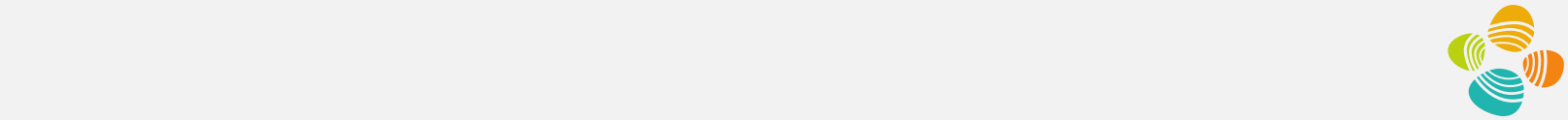
Presence



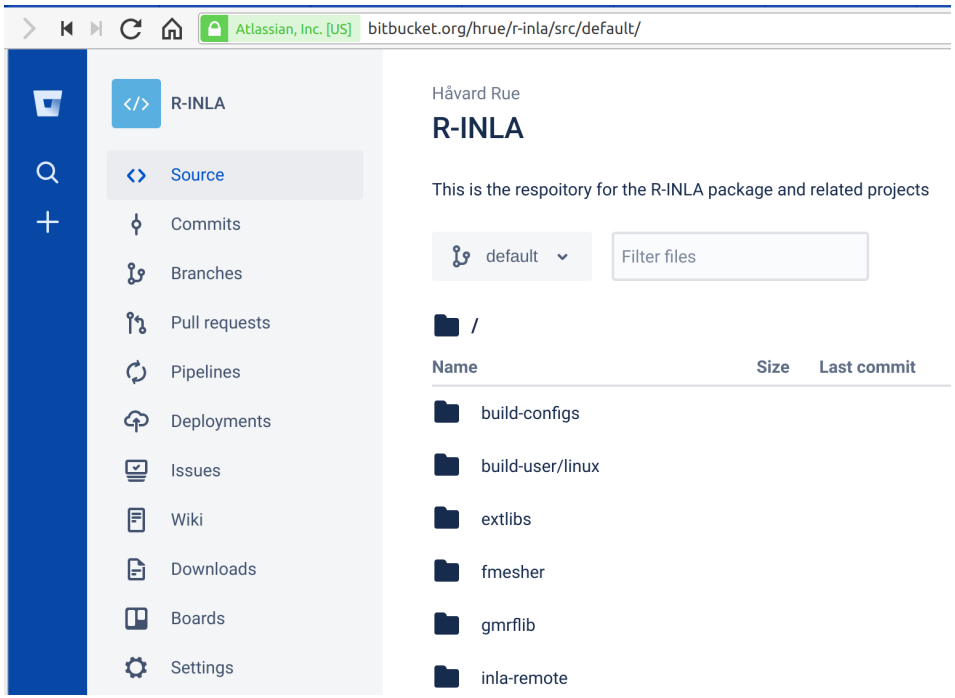
Countries

Country	Users
	25,039 % of Total: 100.00% (25,039)
1. United States	6,474 (25.62%)
2. United Kingdom	2,945 (11.66%)
3. Canada	1,427 (5.65%)
4. Brazil	1,205 (4.77%)
5. Spain	993 (3.93%)
6. Australia	872 (3.45%)
7. France	855 (3.38%)
8. Germany	791 (3.13%)
9. Norway	718 (2.84%)
10. China	706 (2.79%)

The code

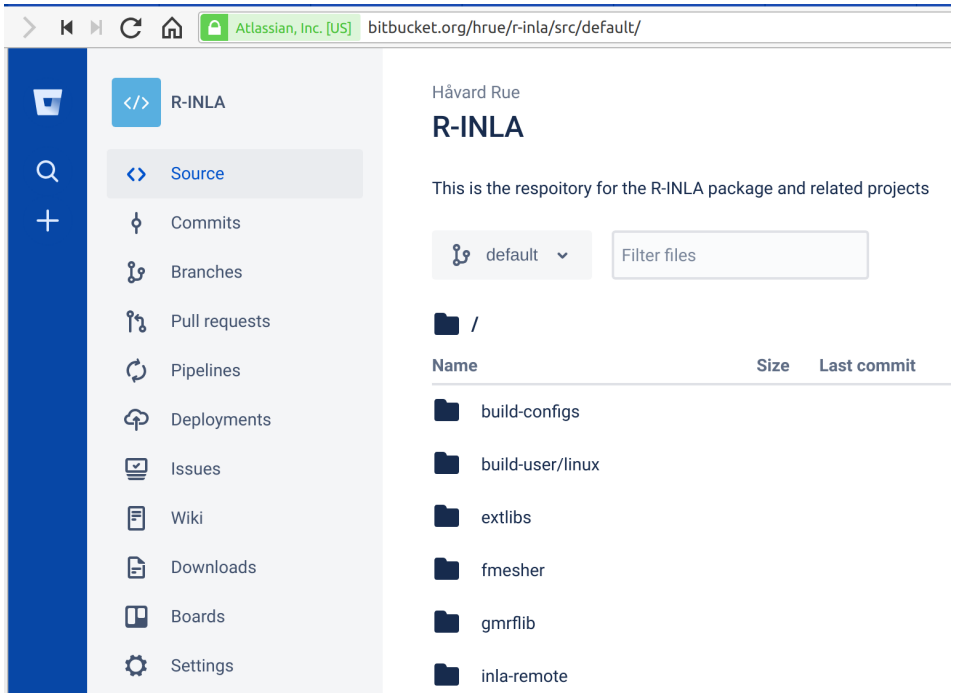


Respository

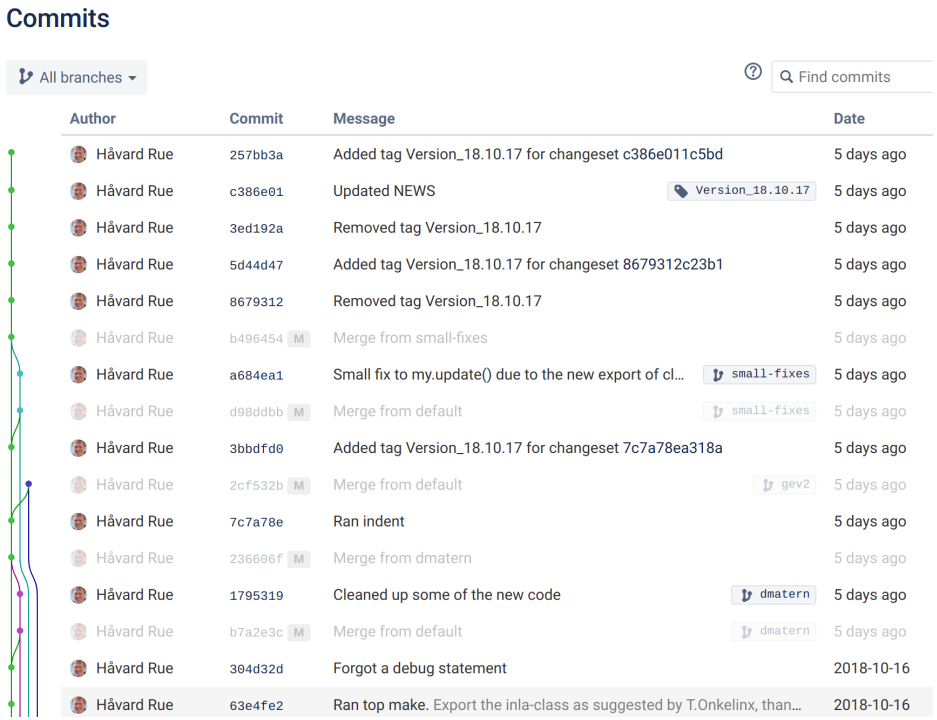


The code

Respository



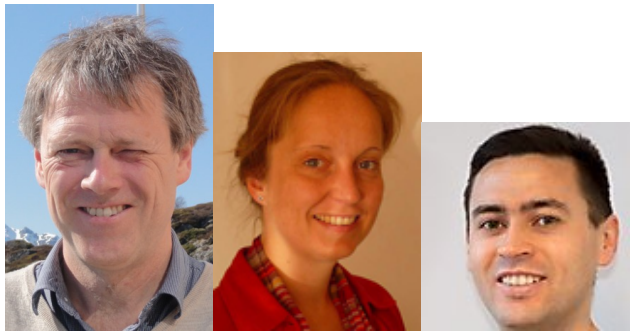
Commits

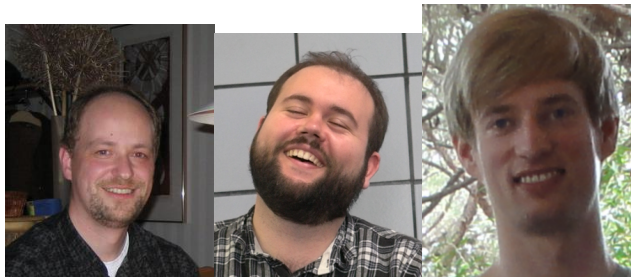


Some of the people, directly and indirectly involved

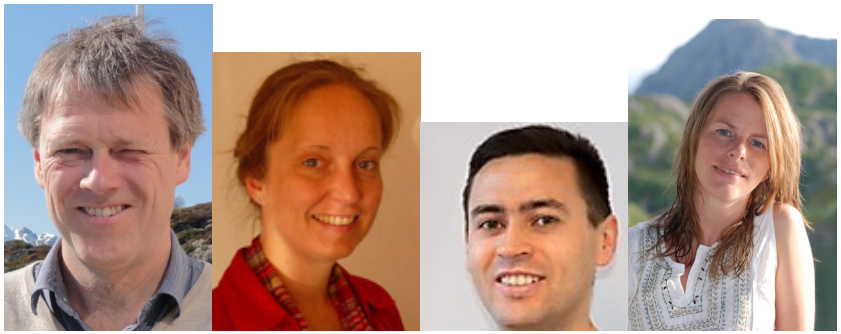
Some of the people, directly and indirectly involved

Some of the people, directly and indirectly involved

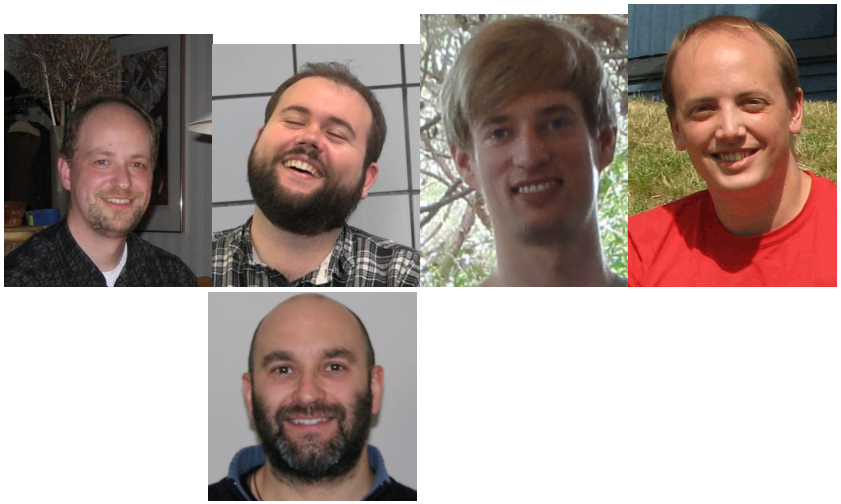




Some of the people, directly and indirectly involved

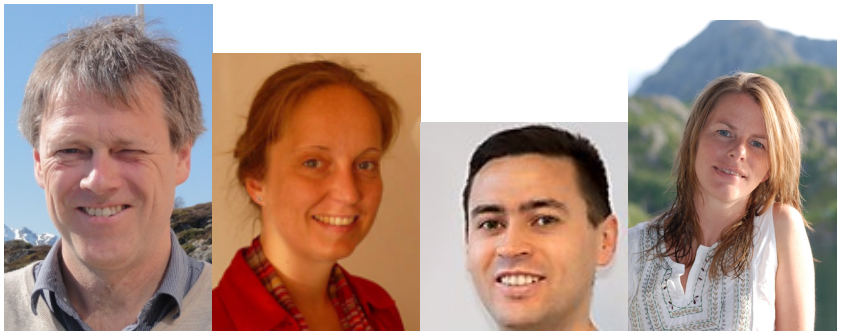


Some of the people, directly and indirectly involved

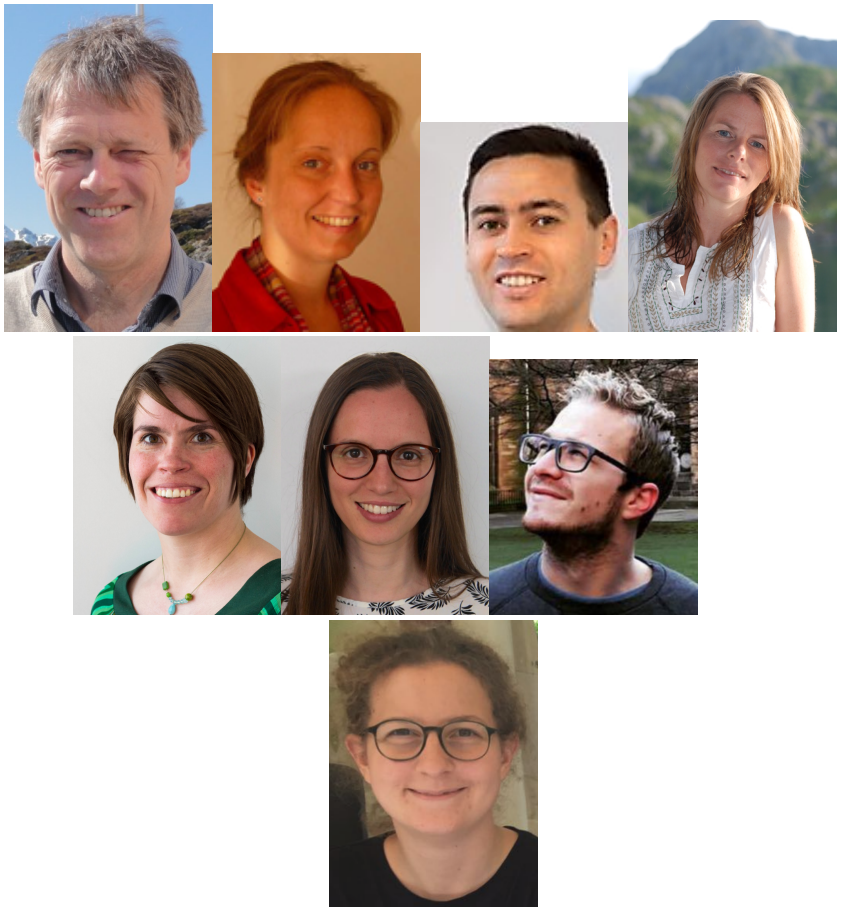


Some of the people, directly and indirectly involved

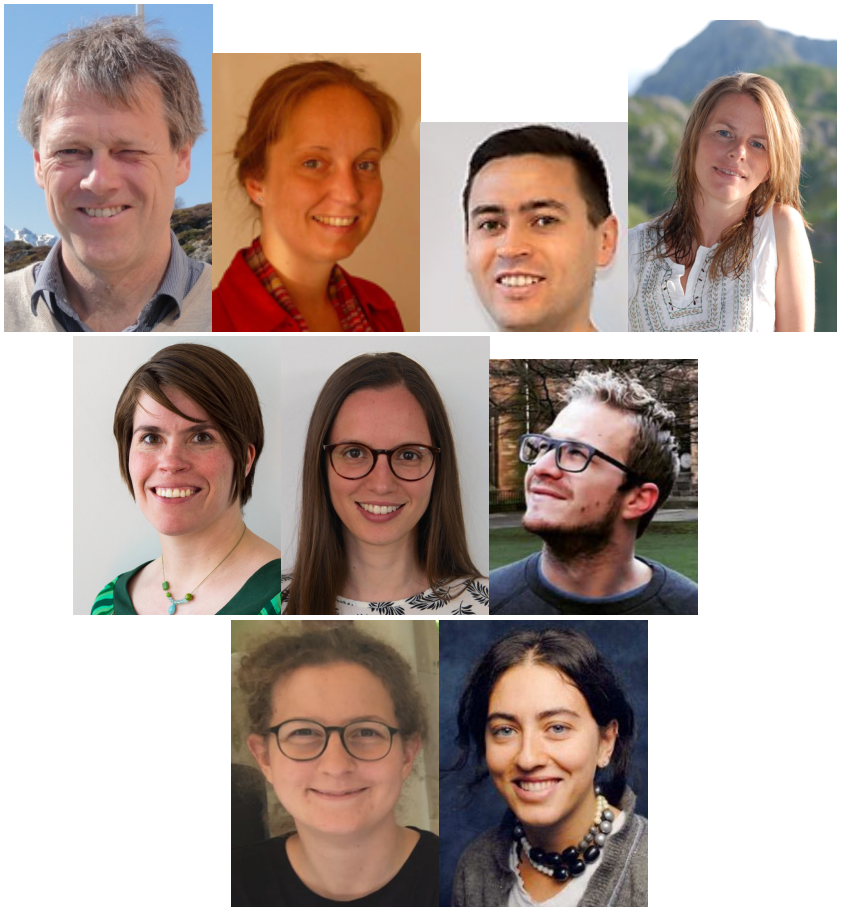
Some of the people, directly and indirectly involved



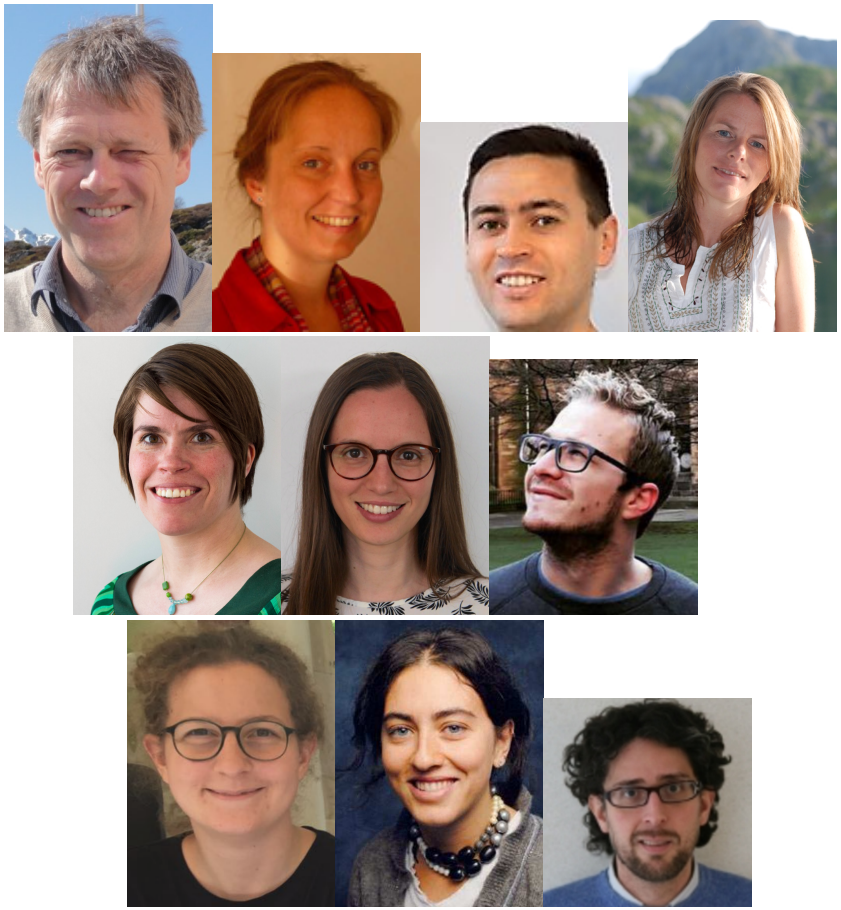
Some of the people, directly and indirectly involved



Some of the people, directly and indirectly involved



Some of the people, directly and indirectly involved



Some of the people, directly and indirectly involved

Part I

Background and Introduction

Background (I)

- The issue of Bayesian computing is not “solved” even though MCMC is available
- Hierarcical models are more difficult for MCMC
- A main obstacle for Bayesian modelling is the issue of “Bayesian computing”
- There are (somewhat) generic tools are available `JAGS`/`stan`, based on MCMC
- Still slower than we would like them to be

Background (I)

- The issue of Bayesian computing is not “solved” even though MCMC is available
- Hierarcical models are more difficult for MCMC
- A main obstacle for Bayesian modelling is the issue of “Bayesian computing”
- There are (somewhat) generic tools are available `JAGS`/`stan`, based on MCMC
- Still slower than we would like them to be

Background (I)

- The issue of Bayesian computing is not “solved” even though MCMC is available
- Hierarcical models are more difficult for MCMC
- A main obstacle for Bayesian modelling is the issue of “Bayesian computing”
- There are (somewhat) generic tools are available `JAGS`/`stan`, based on MCMC
- Still slower than we would like them to be

Background (I)

- The issue of Bayesian computing is not “solved” even though MCMC is available
- Hierarcical models are more difficult for MCMC
- A main obstacle for Bayesian modelling is the issue of “Bayesian computing”
- There are (somewhat) generic tools are available `JAGS`/`stan`, based on MCMC
- Still slower than we would like them to be

Background (I)

- The issue of Bayesian computing is not “solved” even though MCMC is available
- Hierarcical models are more difficult for MCMC
- A main obstacle for Bayesian modelling is the issue of “Bayesian computing”
- There are (somewhat) generic tools are available `JAGS`/`stan`, based on MCMC
- Still slower than we would like them to be

Background (II)

GLM/GAM/GLMM/GAMM/++

- Perhaps the most important class of statistical models
- Many “models” can be cast in to this class without knowing
- No good (enough) MCMC solution around
- Many suggested approaches does not scale well computationally

Background (II)

GLM/GAM/GLMM/GAMM/++

- Perhaps the most important class of statistical models
- Many “models” can be cast in to this class without knowing
- No good (enough) MCMC solution around
- Many suggested approaches does not scale well computationally

Background (II)

GLM/GAM/GLMM/GAMM/++

- Perhaps the most important class of statistical models
- Many “models” can be cast in to this class without knowing
- No good (enough) MCMC solution around
- Many suggested approaches does not scale well computationally

Background (II)

GLM/GAM/GLMM/GAMM/++

- Perhaps the most important class of statistical models
- Many “models” can be cast in to this class without knowing
- No good (enough) MCMC solution around
- Many suggested approaches does not scale well computationally

Bayesian GLM/GAM/GLMM/GAMM/+++

Linear predictor

$$\boldsymbol{\eta} = \mu\mathbf{1} + \boldsymbol{A}\boldsymbol{\beta} + \sum_i \boldsymbol{B}_i \boldsymbol{v}_i + \boldsymbol{\epsilon}$$

where

- $\boldsymbol{A}$: covariates, with fixed effects $\boldsymbol{\beta} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$
- $\boldsymbol{B}_i$: weights, with random effects $\boldsymbol{v}_i \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})$
- $\boldsymbol{\epsilon}_i$: possible Gaussian noise

Observations

$$\boldsymbol{y} \sim \pi(\boldsymbol{y} \mid \boldsymbol{\eta},) = \prod_i \pi(y_i \mid \eta_i)$$

Bayesian GLM/GAM/GLMM/GAMM/+++

Linear predictor

$$\boldsymbol{\eta} = \mu \mathbf{1} + \boldsymbol{A}\boldsymbol{\beta} + \sum_i \boldsymbol{B}_i \boldsymbol{v}_i + \boldsymbol{\epsilon}$$

where

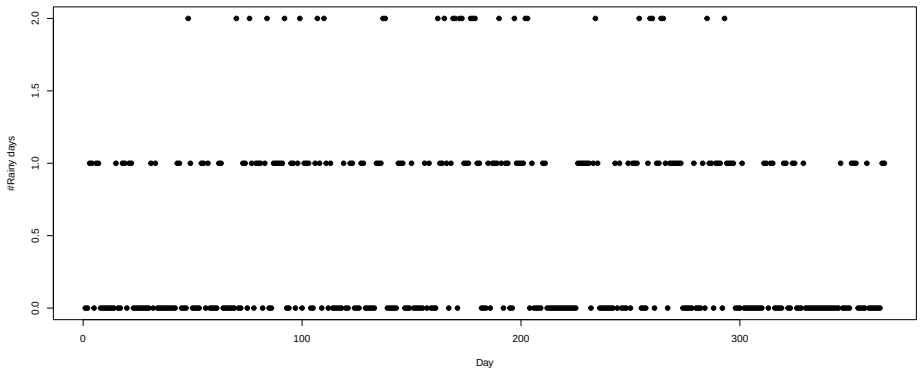
- $\boldsymbol{A}$: covariates, with fixed effects $\boldsymbol{\beta} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$
- $\boldsymbol{B}_i$: weights, with random effects $\boldsymbol{v}_i \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})$
- $\boldsymbol{\epsilon}_i$: possible Gaussian noise

Observations

$$\boldsymbol{y} \sim \pi(\boldsymbol{y} \mid \boldsymbol{\eta},) = \prod_i \pi(y_i \mid \eta_i)$$

Example I

Smoothing of binary/integer time-series



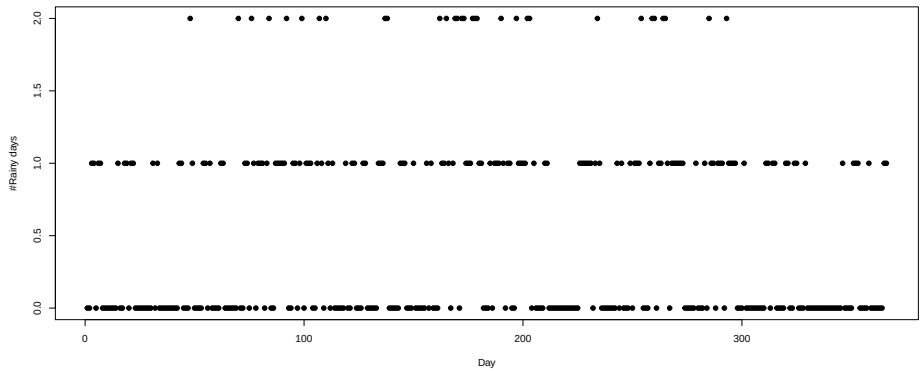
- Data is sequence of 0, 1 and 2's
- Binomial data where probability for a success p_t, depends on time t

$$p_t = \frac{\exp(\eta_t)}{1 + \exp(\eta_t)}$$

- Linear predictor $\eta_t = \mu + \beta c_t + u_t + v_t, \qquad t = 1, \ldots, n$

Example I

Smoothing of binary/integer time-series



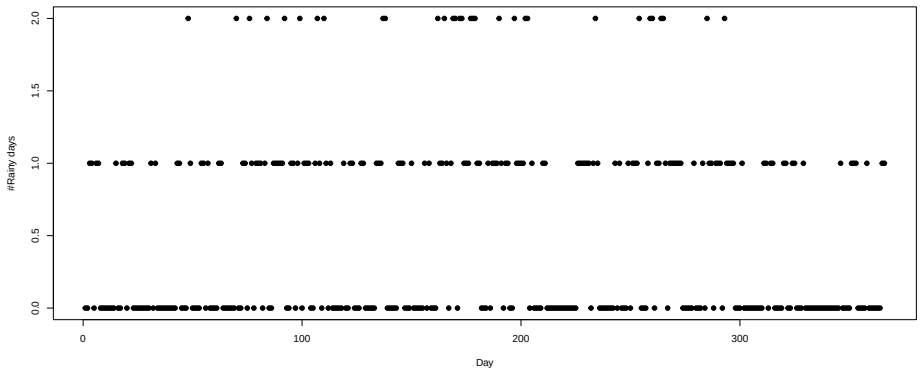
- Data is sequence of 0, 1 and 2's
- Binomial data where probability for a success p_t, depends on time t

$$p_t = \frac{\exp(\eta_t)}{1 + \exp(\eta_t)}$$

- Linear predictor $\eta_t = \mu + \beta c_t + u_t + v_t, \qquad t = 1, \ldots, n$

Example I

Smoothing of binary/integer time-series



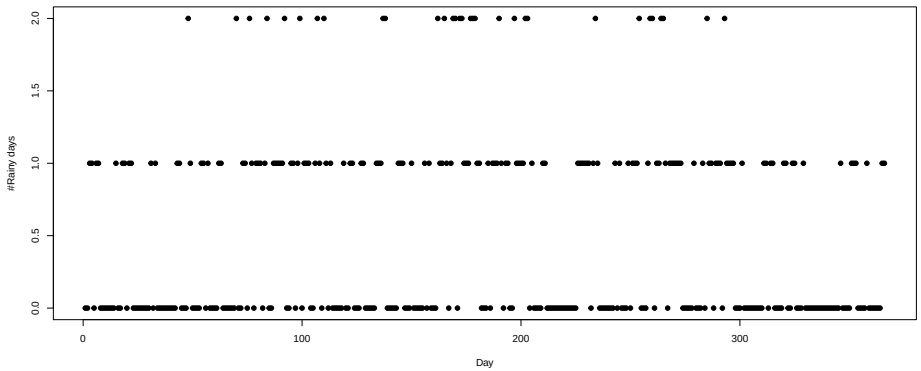
- Data is sequence of $0, 1$ and 2's
- Binomial data where probability for a success p_t, depends on time t

$$p_t = \frac{\exp(\eta_t)}{1 + \exp(\eta_t)}$$

- Linear predictor $\eta_t = \mu + \beta c_t + u_t + v_t, \qquad t = 1, \ldots, n$

Example I

Smoothing of binary/integer time-series



- Data is sequence of $0, 1$ and 2's
- Binomial data where probability for a success p_t, depends on time t

$$p_t = \frac{\exp(\eta_t)}{1 + \exp(\eta_t)}$$

- Linear predictor $\eta_t = \mu + \beta c_t + u_t + v_t, \qquad t = 1, \ldots, n$

Example I

We can reinterpret the model as

$$\begin{aligned} \boldsymbol{\theta} &\sim \pi(\boldsymbol{\theta}) \\ \boldsymbol{x} \mid \boldsymbol{\theta} &\sim \pi(\boldsymbol{x} \mid \boldsymbol{\theta}) = \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}(\boldsymbol{\theta})) \\ \boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta} &\sim \prod_i \pi(y_i \mid \eta_i, \boldsymbol{\theta}) \end{aligned}$$

- $\dim(\boldsymbol{x})$ could be large 10^2-10^5
- $\dim(\boldsymbol{\theta})$ is small 1-5

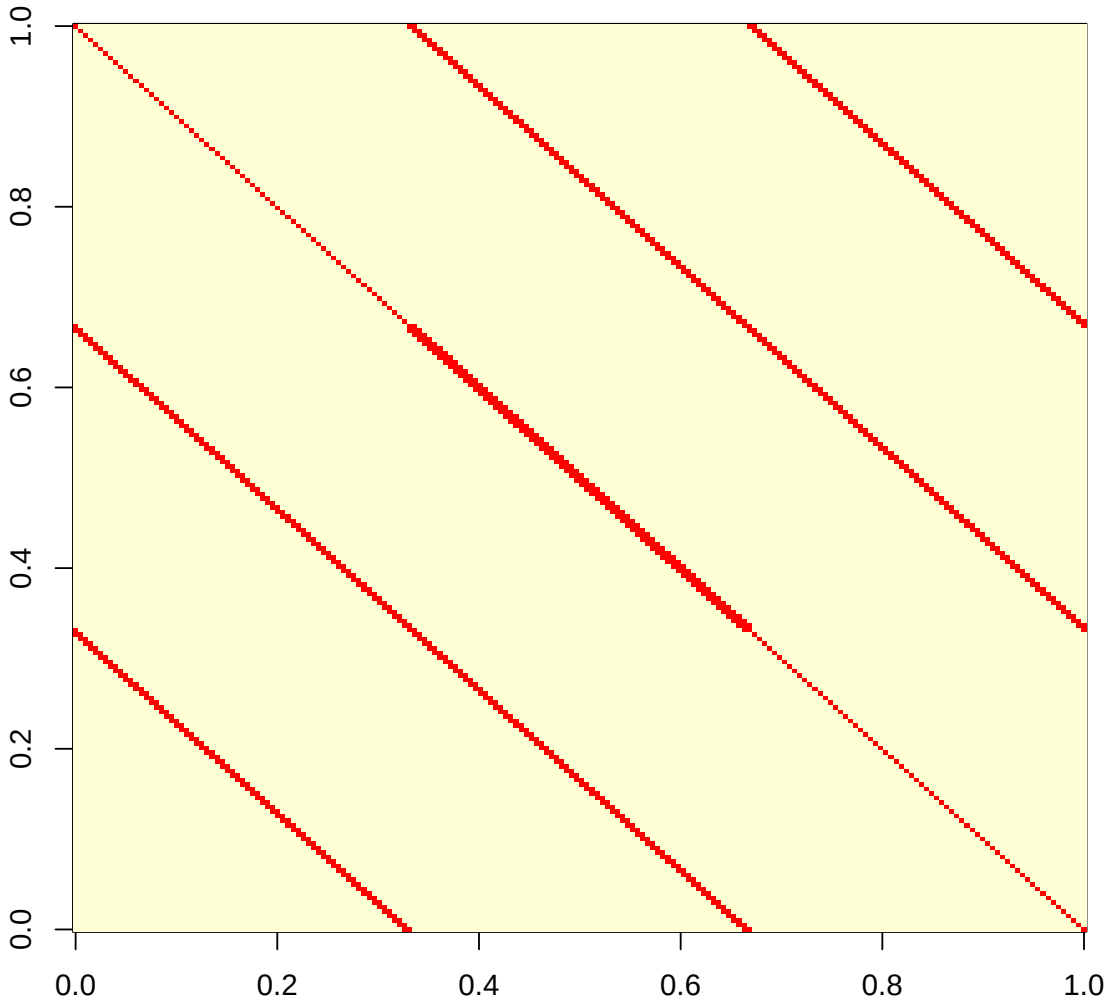
Example I

We can reinterpret the model as

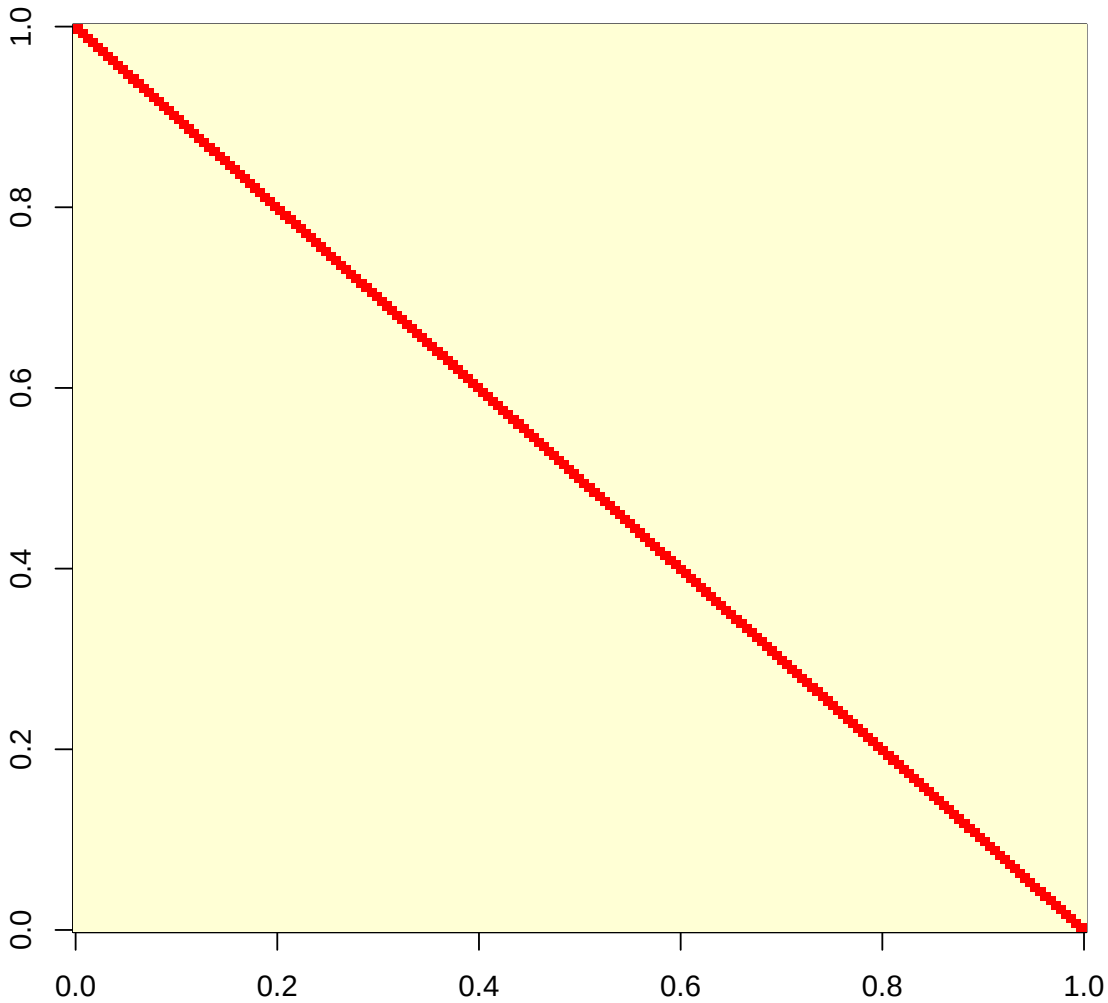
$$\begin{aligned} \boldsymbol{\theta} &\sim \pi(\boldsymbol{\theta}) \\ \boldsymbol{x} \mid \boldsymbol{\theta} &\sim \pi(\boldsymbol{x} \mid \boldsymbol{\theta}) = \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}(\boldsymbol{\theta})) \\ \boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta} &\sim \prod_i \pi(y_i \mid \eta_i, \boldsymbol{\theta}) \end{aligned}$$

- $\dim(\boldsymbol{x})$ could be large 10^2-10^5
- $\dim(\boldsymbol{\theta})$ is small 1-5

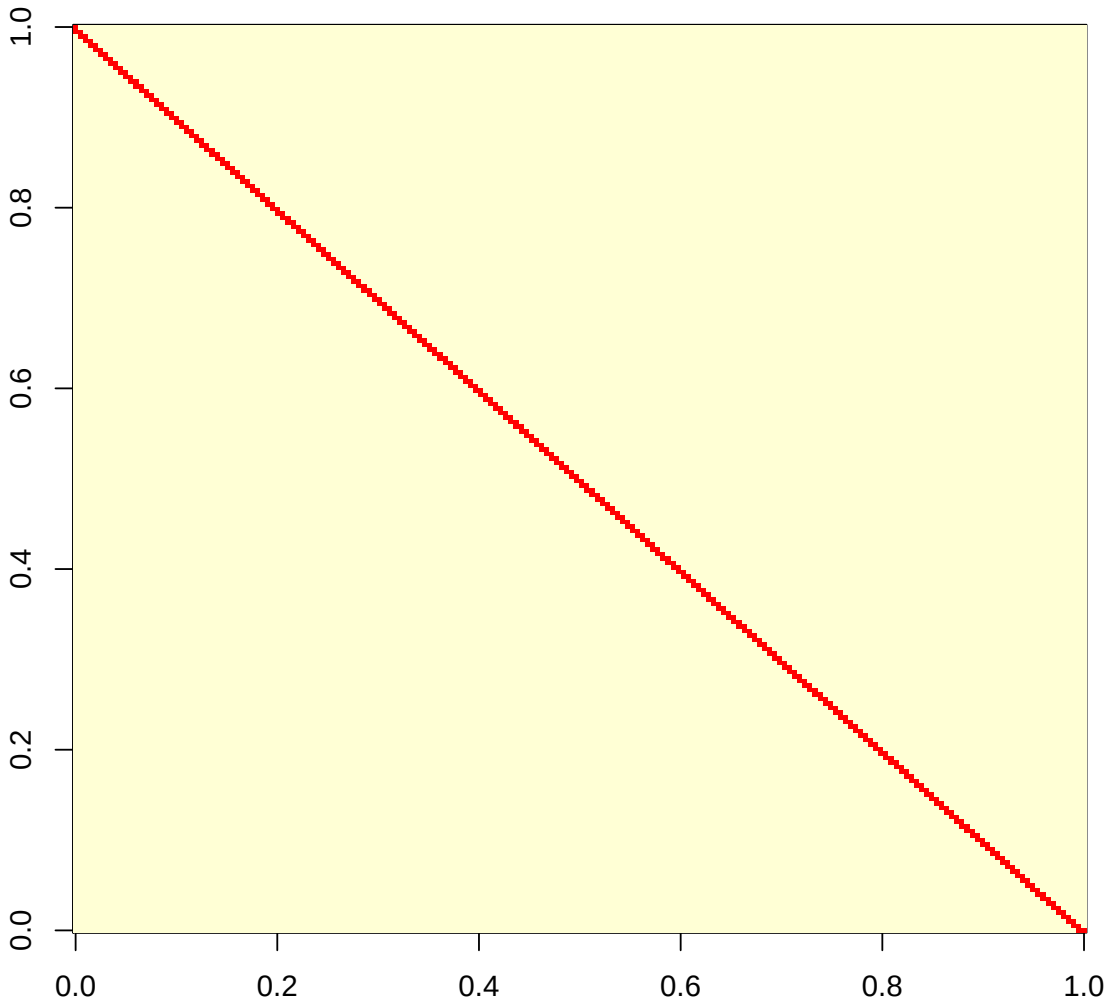
Precision matrix $(\boldsymbol{\eta}, \boldsymbol{u}, \boldsymbol{v}, \mu, \beta)$ $N = 100, M = 5$.



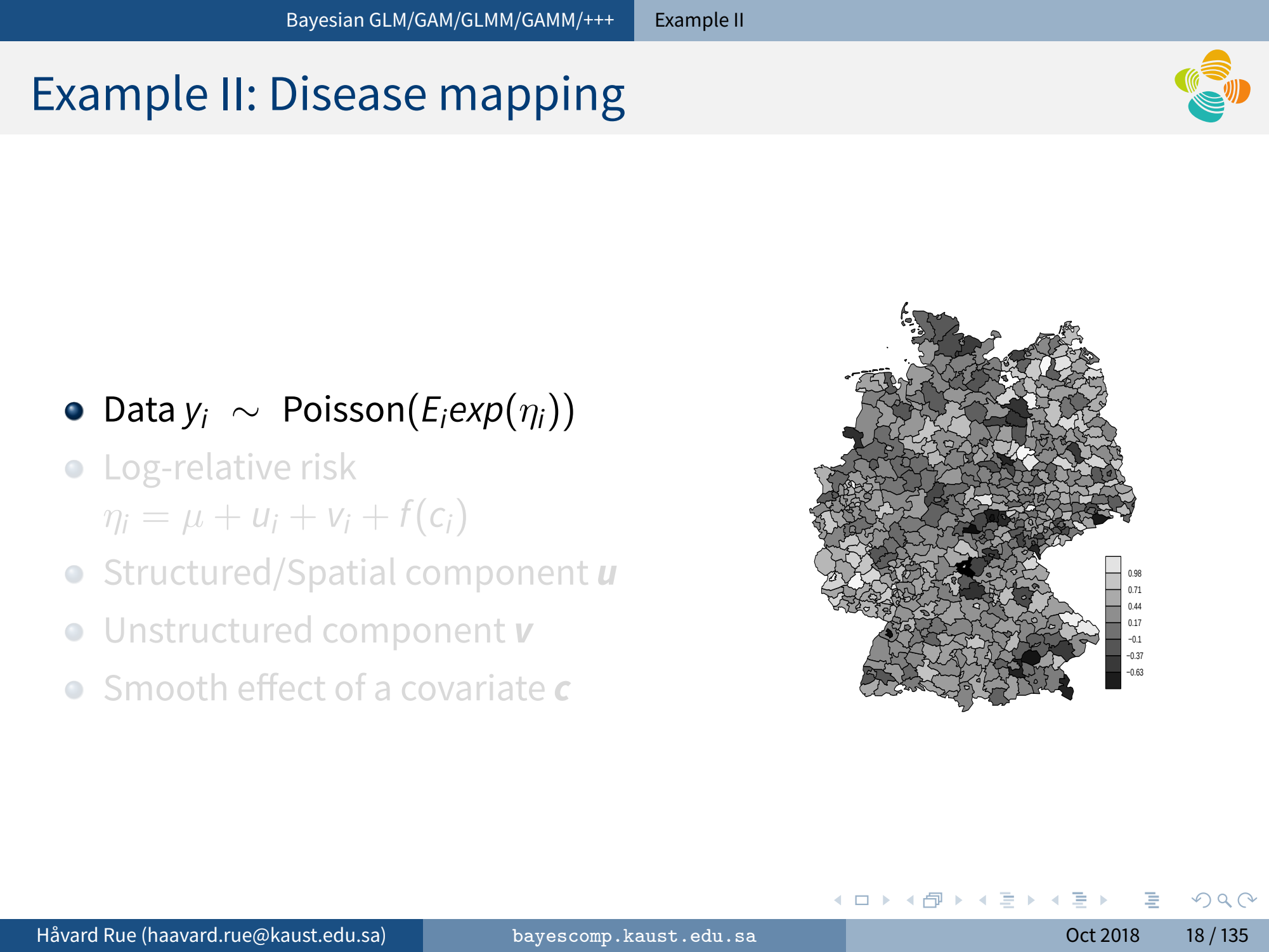
Reordered precision matrix



Cholesky triangle of the reordered precision matrix



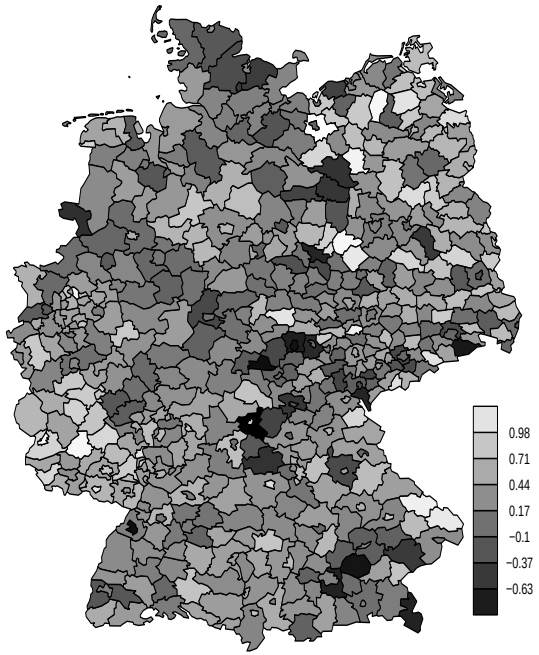
Example II: Disease mapping



- Data $y_i \sim \text{Poisson}(E_i exp(\eta_i))$
- Log-relative risk
  $\eta_i = \mu + u_i + v_i + f(c_i)$
- Structured/Spatial component $\boldsymbol{u}$
- Unstructured component $\boldsymbol{v}$
- Smooth effect of a covariate $\boldsymbol{c}$

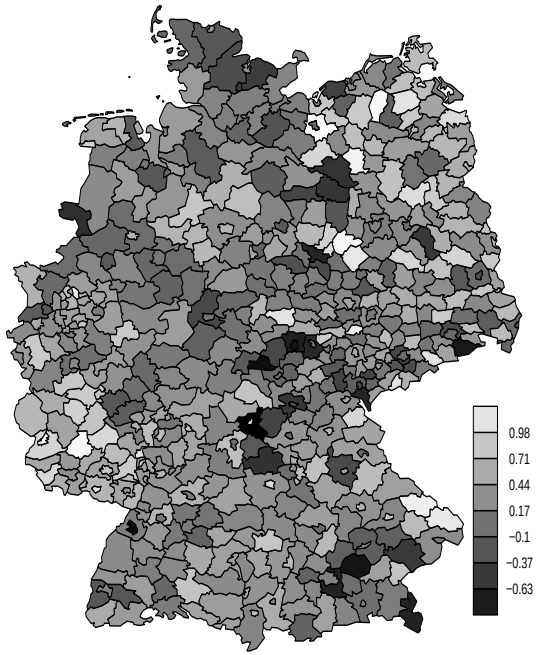
Example II: Disease mapping

- Data $y_i \sim \text{Poisson}(E_i exp(\eta_i))$
- Log-relative risk
  $\eta_i = \mu + u_i + v_i + f(c_i)$
- Structured/Spatial component $\boldsymbol{u}$
- Unstructured component $\boldsymbol{v}$
- Smooth effect of a covariate $\boldsymbol{c}$



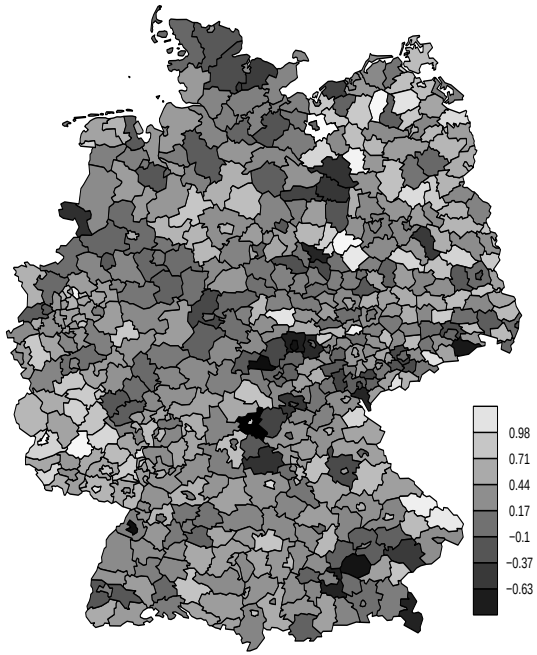
Example II: Disease mapping

- Data $y_i \sim$ Poisson$(E_i exp(\eta_i))$
- Log-relative risk
  $\eta_i = \mu + u_i + v_i + f(c_i)$
- Structured/Spatial component $\boldsymbol{u}$
- Unstructured component $\boldsymbol{v}$
- Smooth effect of a covariate $\boldsymbol{c}$



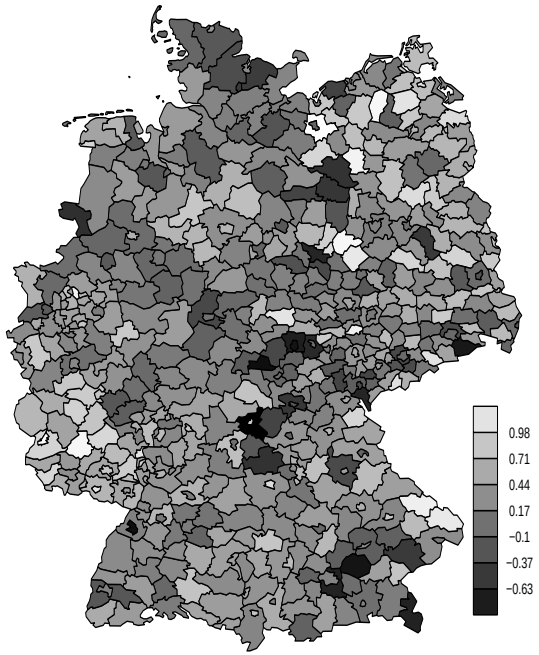
Example II: Disease mapping

- Data $y_i \sim \text{Poisson}(E_i exp(\eta_i))$
- Log-relative risk
  $\eta_i = \mu + u_i + v_i + f(c_i)$
- Structured/Spatial component $\boldsymbol{u}$
- Unstructured component $\boldsymbol{v}$
- Smooth effect of a covariate $\boldsymbol{c}$

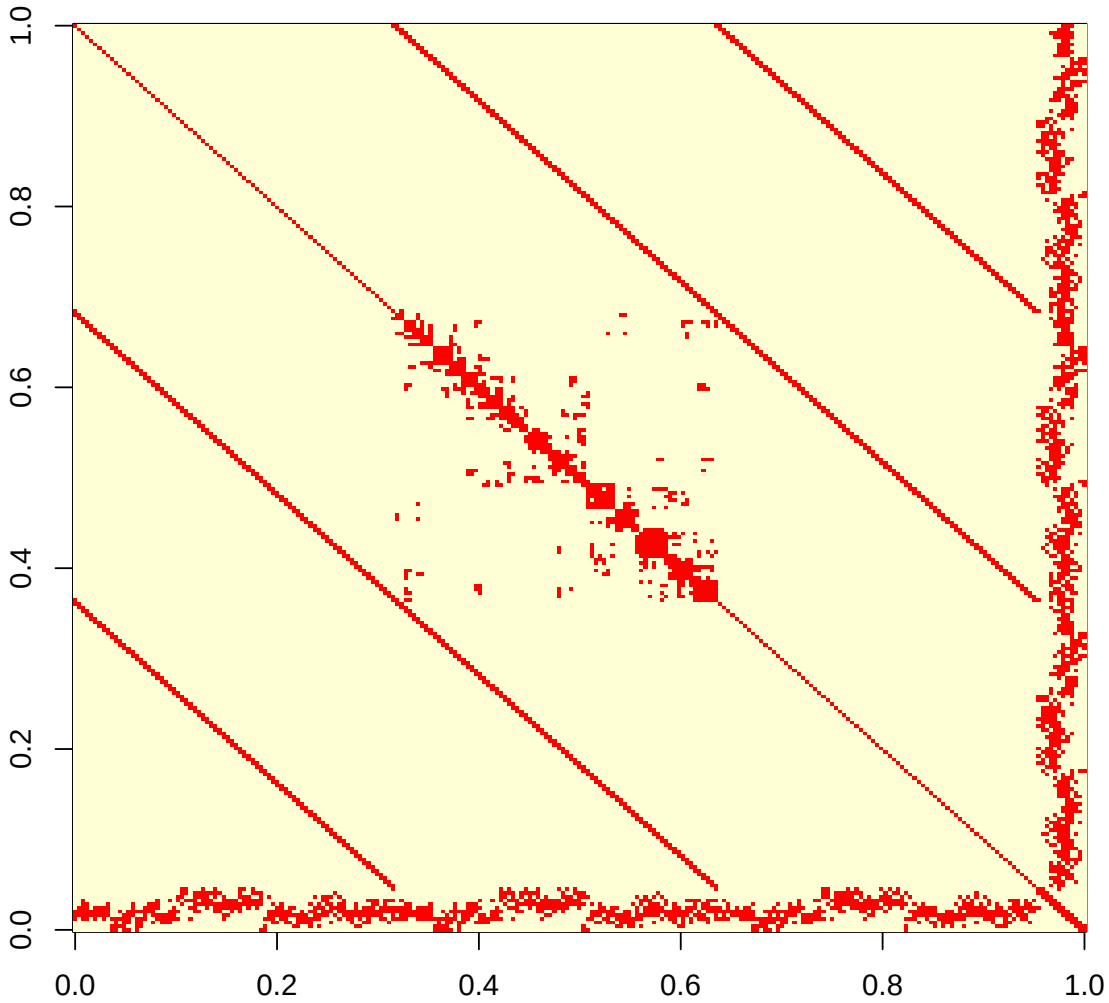


Example II: Disease mapping

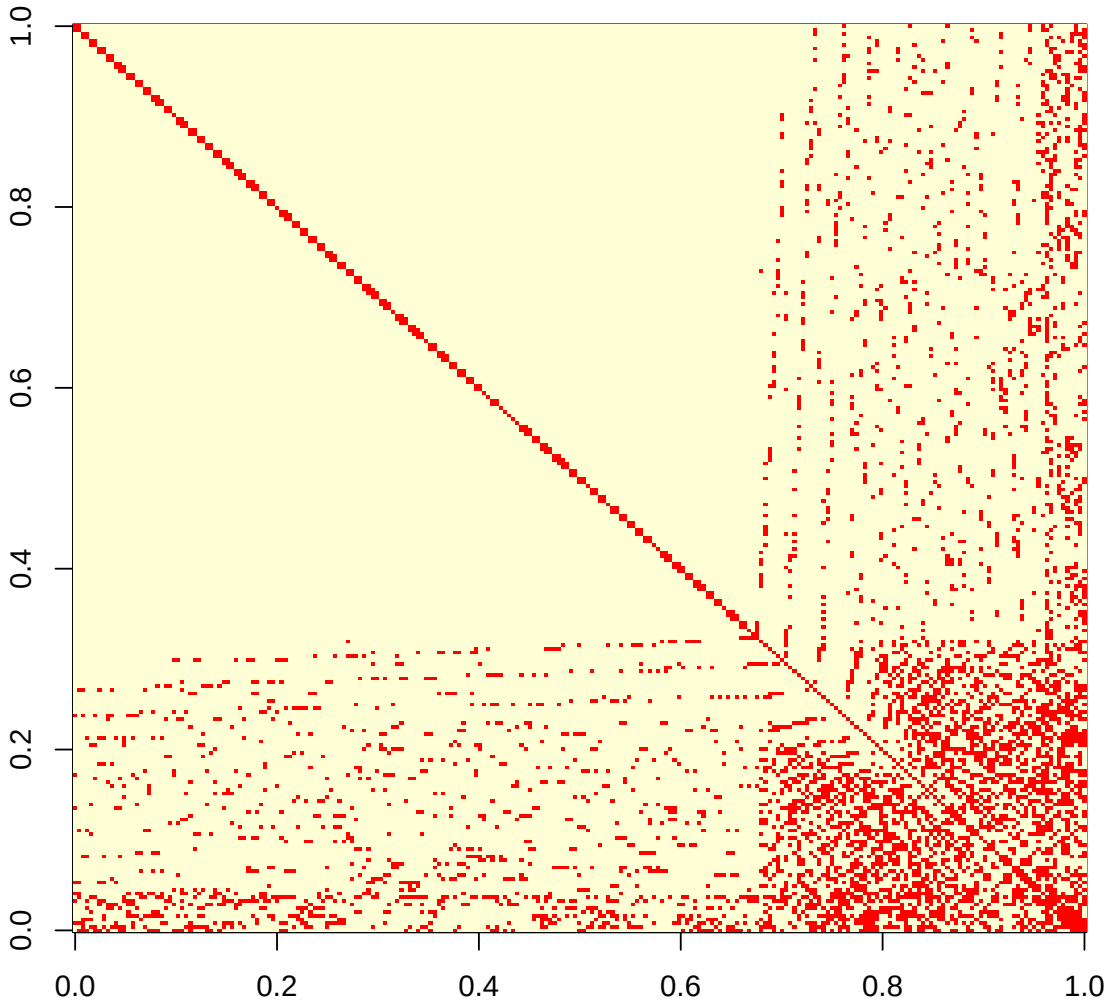
- Data $y_i \sim \text{Poisson}(E_i exp(\eta_i))$
- Log-relative risk
  $\eta_i = \mu + u_i + v_i + f(c_i)$
- Structured/Spatial component $\boldsymbol{u}$
- Unstructured component $\boldsymbol{v}$
- Smooth effect of a covariate $\boldsymbol{c}$



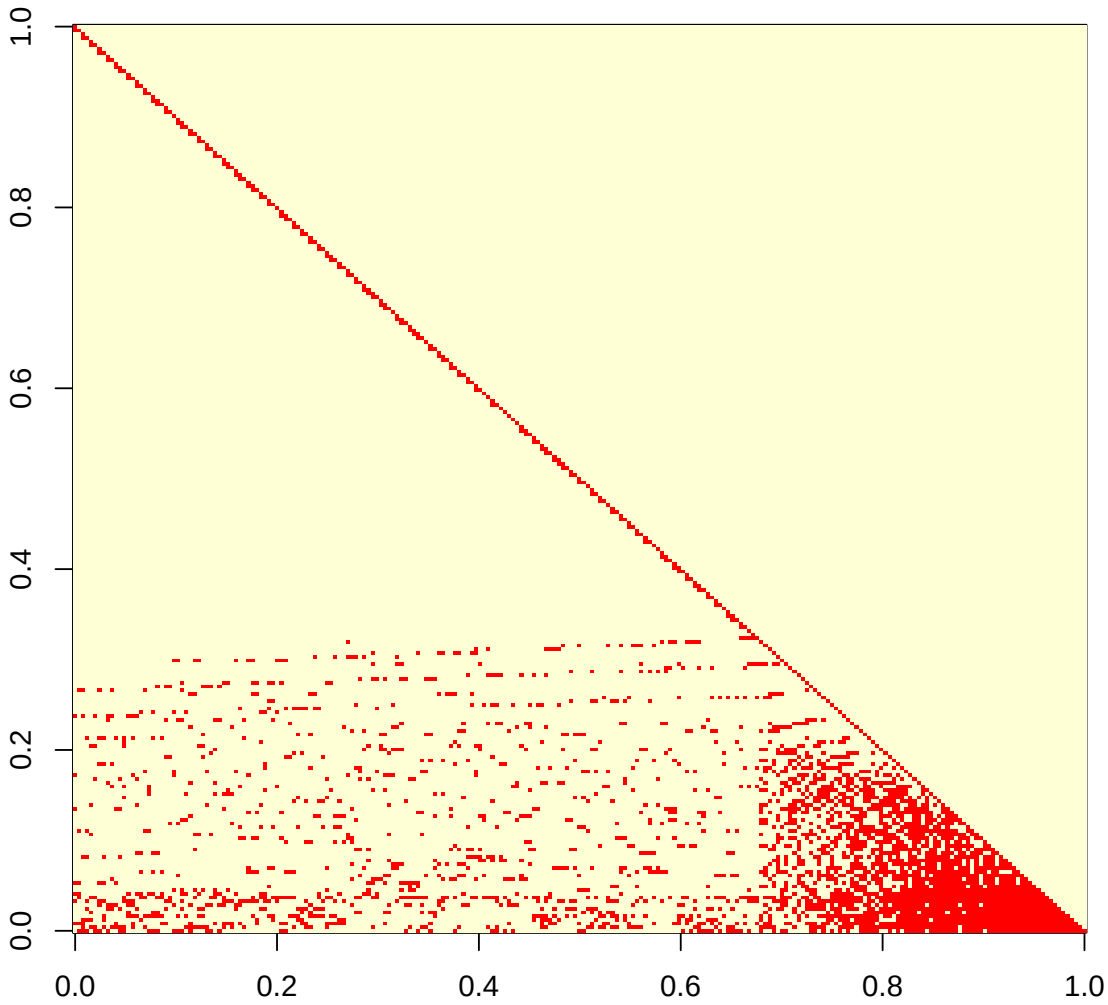
Precision matrix $(\boldsymbol{\eta}, \boldsymbol{u}, \boldsymbol{v}, \mu, \boldsymbol{f})$



Reordered precision matrix



Cholesky triangle of the reordered precision matrix



Example II

We can reinterpret the model as

$$\begin{aligned} \boldsymbol{\theta} &\sim \pi(\boldsymbol{\theta}) \\ \boldsymbol{x} \mid \boldsymbol{\theta} &\sim \pi(\boldsymbol{x} \mid \boldsymbol{\theta}) = \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}(\boldsymbol{\theta})) \\ \boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta} &\sim \prod_i \pi(y_i \mid \eta_i, \boldsymbol{\theta}) \end{aligned}$$

- $\dim(\boldsymbol{x})$ could be large 10^2-10^5
- $\dim(\boldsymbol{\theta})$ is small 1-5

What we have learned so far (I)

This model-construct

$$\begin{aligned} \boldsymbol{\theta} &\sim \pi(\boldsymbol{\theta}) \\ \boldsymbol{x} \mid \boldsymbol{\theta} &\sim \pi(\boldsymbol{x} \mid \boldsymbol{\theta}) = \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}(\boldsymbol{\theta})) \\ \boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta} &\sim \prod_i \pi(y_i \mid \eta_i, \boldsymbol{\theta}) \end{aligned}$$

occurs in seemingly unrelated, statistical models.

Latent Gaussian models!

Further Examples

- Dynamic linear models
- Stochastic volatility
- Generalised linear (mixed) models
- Generalised additive (mixed) models
- Measurement error models
- Spline smoothing
- Semiparametric regression
- Space-varying (semiparametric) regression models
- Disease mapping
- Log-Gaussian Cox-processes
- Model-based geostatistics (*)
- Spatio-temporal models
- Survival analysis
- Joint survival/longitudional models
- +++

Further Examples

- Dynamic linear models
- Stochastic volatility
- Generalised linear (mixed) models
- Generalised additive (mixed) models
- Measurement error models
- Spline smoothing
- Semiparametric regression
- Space-varying (semiparametric) regression models
- Disease mapping
- Log-Gaussian Cox-processes
- Model-based geostatistics (*)
- Spatio-temporal models
- Survival analysis
- Joint survival/longitudinal models
- +++

Further Examples

- Dynamic linear models
- Stochastic volatility
- Generalised linear (mixed) models
- Generalised additive (mixed) models
- Measurement error models
- Spline smoothing
- Semiparametric regression
- Space-varying (semiparametric) regression models
- Disease mapping
- Log-Gaussian Cox-processes
- Model-based geostatistics (*)
- Spatio-temporal models
- Survival analysis
- Joint survival/longitudional models
- +++

Further Examples

- Dynamic linear models
- Stochastic volatility
- Generalised linear (mixed) models
- Generalised additive (mixed) models
- Measurement error models
- Spline smoothing
- Semiparametric regression
- Space-varying (semiparametric) regression models
- Disease mapping
- Log-Gaussian Cox-processes
- Model-based geostatistics (*)
- Spatio-temporal models
- Survival analysis
- Joint survival/longitudinal models
- +++

Further Examples

- Dynamic linear models
- Stochastic volatility
- Generalised linear (mixed) models
- Generalised additive (mixed) models
- Measurement error models
- Spline smoothing
- Semiparametric regression
- Space-varying (semiparametric) regression models
- Disease mapping
- Log-Gaussian Cox-processes
- Model-based geostatistics (*)
- Spatio-temporal models
- Survival analysis
- Joint survival/longitudinal models
- +++

Further Examples

- Dynamic linear models
- Stochastic volatility
- Generalised linear (mixed) models
- Generalised additive (mixed) models
- Measurement error models
- Spline smoothing
- Semiparametric regression
- Space-varying (semiparametric) regression models
- Disease mapping
- Log-Gaussian Cox-processes
- Model-based geostatistics (*)
- Spatio-temporal models
- Survival analysis
- Joint survival/longitudional models
- +++

Further Examples

- Dynamic linear models
- Stochastic volatility
- Generalised linear (mixed) models
- Generalised additive (mixed) models
- Measurement error models
- Spline smoothing
- Semiparametric regression
- Space-varying (semiparametric) regression models
- Disease mapping
- Log-Gaussian Cox-processes
- Model-based geostatistics (*)
- Spatio-temporal models
- Survival analysis
- Joint survival/longitudional models
- +++

Further Examples

- Dynamic linear models
- Stochastic volatility
- Generalised linear (mixed) models
- Generalised additive (mixed) models
- Measurement error models
- Spline smoothing
- Semiparametric regression
- Space-varying (semiparametric) regression models
- Disease mapping
- Log-Gaussian Cox-processes
- Model-based geostatistics (*)
- Spatio-temporal models
- Survival analysis
- Joint survival/longitudinal models
- +++

Further Examples

- Dynamic linear models
- Stochastic volatility
- Generalised linear (mixed) models
- Generalised additive (mixed) models
- Measurement error models
- Spline smoothing
- Semiparametric regression
- Space-varying (semiparametric) regression models
- Disease mapping
- Log-Gaussian Cox-processes
- Model-based geostatistics (*)
- Spatio-temporal models
- Survival analysis
- Joint survival/longitudional models
- +++

Further Examples

- Dynamic linear models
- Stochastic volatility
- Generalised linear (mixed) models
- Generalised additive (mixed) models
- Measurement error models
- Spline smoothing
- Semiparametric regression
- Space-varying (semiparametric) regression models
- Disease mapping
- Log-Gaussian Cox-processes
- Model-based geostatistics (*)
- Spatio-temporal models
- Survival analysis
- Joint survival/longitudinal models
- +++

Further Examples

- Dynamic linear models
- Stochastic volatility
- Generalised linear (mixed) models
- Generalised additive (mixed) models
- Measurement error models
- Spline smoothing
- Semiparametric regression
- Space-varying (semiparametric) regression models
- Disease mapping
- Log-Gaussian Cox-processes
- Model-based geostatistics (*)
- Spatio-temporal models
- Survival analysis
- Joint survival/longitudinal models
- +++

Further Examples

- Dynamic linear models
- Stochastic volatility
- Generalised linear (mixed) models
- Generalised additive (mixed) models
- Measurement error models
- Spline smoothing
- Semiparametric regression
- Space-varying (semiparametric) regression models
- Disease mapping
- Log-Gaussian Cox-processes
- Model-based geostatistics (*)
- Spatio-temporal models
- Survival analysis
- Joint survival/longitudional models
- +++

Further Examples

- Dynamic linear models
- Stochastic volatility
- Generalised linear (mixed) models
- Generalised additive (mixed) models
- Measurement error models
- Spline smoothing
- Semiparametric regression
- Space-varying (semiparametric) regression models
- Disease mapping
- Log-Gaussian Cox-processes
- Model-based geostatistics (*)
- Spatio-temporal models
- Survival analysis
- Joint survival/longitudional models
- +++

Further Examples

- Dynamic linear models
- Stochastic volatility
- Generalised linear (mixed) models
- Generalised additive (mixed) models
- Measurement error models
- Spline smoothing
- Semiparametric regression
- Space-varying (semiparametric) regression models
- Disease mapping
- Log-Gaussian Cox-processes
- Model-based geostatistics (*)
- Spatio-temporal models
- Survival analysis
- Joint survival/longitudional models
- +++

Further Examples

- Dynamic linear models
- Stochastic volatility
- Generalised linear (mixed) models
- Generalised additive (mixed) models
- Measurement error models
- Spline smoothing
- Semiparametric regression
- Space-varying (semiparametric) regression models
- Disease mapping
- Log-Gaussian Cox-processes
- Model-based geostatistics (*)
- Spatio-temporal models
- Survival analysis
- Joint survival/longitudinal models
- +++

What we have learned so far (II)

The precision matrix of the latent field

$$\mathbf{Q}(\theta) = \mathbf{\Sigma}(\theta)^{-1}$$

is sparse and this will play a key role!

Two important benefits

- Building models through conditioning ("hierarchical models")
- Computational

What we have learned so far (II)

The precision matrix of the latent field

$$\boldsymbol{Q}(\boldsymbol{\theta}) = \boldsymbol{\Sigma}(\boldsymbol{\theta})^{-1}$$

is sparse and this will play a key role!

Two important benefits

- Building models through conditioning ("hierarchical models")
- Computational

Building models through conditioning

If

- $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_x^{-1})$
- $\boldsymbol{y}|\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{x}, \boldsymbol{Q}_y^{-1})$

then

$$\boldsymbol{Q}_{(\boldsymbol{x},\boldsymbol{y})} = \begin{bmatrix} \boldsymbol{Q}_x + \boldsymbol{Q}_y & -\boldsymbol{Q}_y \\ -\boldsymbol{Q}_y & \boldsymbol{Q}_y \end{bmatrix}$$

Not so nice expressions using the Covariance-matrix

Building models through conditioning

If

- $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_x^{-1})$
- $\boldsymbol{y}|\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{x}, \boldsymbol{Q}_y^{-1})$

then

$$\boldsymbol{Q}_{(\boldsymbol{x},\boldsymbol{y})} = \begin{bmatrix} \boldsymbol{Q}_x + \boldsymbol{Q}_y & -\boldsymbol{Q}_y \\ -\boldsymbol{Q}_y & \boldsymbol{Q}_y \end{bmatrix}$$

Not so nice expressions using the Covariance-matrix

Building models through conditioning

If

- $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_x^{-1})$
- $\boldsymbol{y}|\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{x}, \boldsymbol{Q}_y^{-1})$

then

$$\boldsymbol{Q}_{(\boldsymbol{x},\boldsymbol{y})} = \begin{bmatrix} \boldsymbol{Q}_x + \boldsymbol{Q}_y & -\boldsymbol{Q}_y \\ -\boldsymbol{Q}_y & \boldsymbol{Q}_y \end{bmatrix}$$

Not so nice expressions using the Covariance-matrix

Computational benefits

- Models we have seen gives a sparse precision matrix
- These are much faster to compute with, than dense matrices
- Special case: Kalman-filter algorithms

Tasks:

- Factorize $\boldsymbol{Q}$ into $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$ (Cholesky)
- Solve $\boldsymbol{Q}\boldsymbol{x} = \boldsymbol{b}$, $\boldsymbol{L}\boldsymbol{x} = \boldsymbol{b}$ or $\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{b}$
- Compute diag$(\boldsymbol{Q}^{-1})$

Computational benefits

- Models we have seen gives a sparse precision matrix
- These are much faster to compute with, than dense matrices
- Special case: Kalman-filter algorithms

Tasks:

- Factorize $\boldsymbol{Q}$ into $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$ (Cholesky)
- Solve $\boldsymbol{Q}\boldsymbol{x} = \boldsymbol{b}$, $\boldsymbol{L}\boldsymbol{x} = \boldsymbol{b}$ or $\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{b}$
- Compute $\text{diag}(\boldsymbol{Q}^{-1})$

Computational benefits

- Models we have seen gives a sparse precision matrix
- These are much faster to compute with, than dense matrices
- Special case: Kalman-filter algorithms

Tasks:

- Factorize $\boldsymbol{Q}$ into $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$ (Cholesky)
- Solve $\boldsymbol{Q}\boldsymbol{x} = \boldsymbol{b}$, $\boldsymbol{L}\boldsymbol{x} = \boldsymbol{b}$ or $\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{b}$
- Compute $\text{diag}(\boldsymbol{Q}^{-1})$

Computational benefits

- Models we have seen gives a sparse precision matrix
- These are much faster to compute with, than dense matrices
- Special case: Kalman-filter algorithms

Tasks:

- Factorize $\boldsymbol{Q}$ into $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$ (Cholesky)
- Solve $\boldsymbol{Q}\boldsymbol{x} = \boldsymbol{b}$, $\boldsymbol{L}\boldsymbol{x} = \boldsymbol{b}$ or $\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{b}$
- Compute $\text{diag}(\boldsymbol{Q}^{-1})$

Computational benefits

- Models we have seen gives a sparse precision matrix
- These are much faster to compute with, than dense matrices
- Special case: Kalman-filter algorithms

Tasks:

- Factorize $\boldsymbol{Q}$ into $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$ (Cholesky)
- Solve $\boldsymbol{Q}\boldsymbol{x} = \boldsymbol{b}$, $\boldsymbol{L}\boldsymbol{x} = \boldsymbol{b}$ or $\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{b}$
- Compute $\text{diag}(\boldsymbol{Q}^{-1})$

Computational benefits

- Models we have seen gives a sparse precision matrix
- These are much faster to compute with, than dense matrices
- Special case: Kalman-filter algorithms

Tasks:

- Factorize $\boldsymbol{Q}$ into $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$ (Cholesky)
- Solve $\boldsymbol{Q}\boldsymbol{x} = \boldsymbol{b}$, $\boldsymbol{L}\boldsymbol{x} = \boldsymbol{b}$ or $\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{b}$
- Compute $\text{diag}(\boldsymbol{Q}^{-1})$

Computational benefits

- Models we have seen gives a sparse precision matrix
- These are much faster to compute with, than dense matrices
- Special case: Kalman-filter algorithms

Tasks:

- Factorize $\boldsymbol{Q}$ into $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$ (Cholesky)
- Solve $\boldsymbol{Q}\boldsymbol{x} = \boldsymbol{b}$, $\boldsymbol{L}\boldsymbol{x} = \boldsymbol{b}$ or $\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{b}$
- Compute diag$(\boldsymbol{Q}^{-1})$

Numerical algorithms for sparse matrices: scaling properties

Cholesky factorization of "sparse" SPD[1] matrix

- Time: $\mathcal{O}(n)$
- Space: $\mathcal{O}(n^{3/2})$
- Space-time: $\mathcal{O}(n^2)$

This is to be compared with general $\mathcal{O}(n^3)$ algorithm for the Cholesky factorization of a SPD dense matrix.

[1] Symmetric and positive definite

Numerical algorithms for sparse matrices: scaling properties

Cholesky factorization of "sparse" SPD[1] matrix

- Time: $\mathcal{O}(n)$
- Space: $\mathcal{O}(n^{3/2})$
- Space-time: $\mathcal{O}(n^2)$

This is to be compared with general $\mathcal{O}(n^3)$ algorithm for the Cholesky factorization of a SPD dense matrix.

[1] Symmetric and positive definite

Numerical algorithms for sparse matrices: scaling properties

Cholesky factorization of "sparse" SPD[1] matrix

- Time: $\mathcal{O}(n)$
- Space: $\mathcal{O}(n^{3/2})$
- Space-time: $\mathcal{O}(n^2)$

This is to be compared with general $\mathcal{O}(n^3)$ algorithm for the Cholesky factorization of a SPD dense matrix.

[1] Symmetric and positive definite

Numerical algorithms for sparse matrices: scaling properties

Cholesky factorization of "sparse" SPD[1] matrix

- Time: $\mathcal{O}(n)$
- Space: $\mathcal{O}(n^{3/2})$
- Space-time: $\mathcal{O}(n^2)$

This is to be compared with general $\mathcal{O}(n^3)$ algorithm for the Cholesky factorization of a SPD dense matrix.

[1] Symmetric and positive definite

Numerical algorithms for sparse matrices: scaling properties

Cholesky factorization of "sparse" SPD[1] matrix

- Time: $\mathcal{O}(n)$
- Space: $\mathcal{O}(n^{3/2})$
- Space-time: $\mathcal{O}(n^2)$

This is to be compared with general $\mathcal{O}(n^3)$ algorithm for the Cholesky factorization of a SPD dense matrix.

[1] Symmetric and positive definite

Gaussian Markov random fields

- Gaussians with a sparse precision matrix are called *Gaussian Markov random fields* (GMRFs)
- Good computational properties through numerical algorithms for sparse matrices
- Very useful in other settings as well

Gaussian Markov random fields

- Gaussians with a sparse precision matrix are called *Gaussian Markov random fields* (GMRFs)
- Good computational properties through numerical algorithms for sparse matrices
- Very useful in other settings as well

Gaussian Markov random fields

- Gaussians with a sparse precision matrix are called *Gaussian Markov random fields* (GMRFs)
- Good computational properties through numerical algorithms for sparse matrices
- Very useful in other settings as well

Summary

Three main ingredients in INLA

- Gaussian Markov random fields
- Latent Gaussian models
- Laplace approximations

which together (and ++++...) gives a very very nice tool for Bayesian inference

- quick
- accurate (relative error)
- good scaling properties
- +++

Summary

Three main ingredients in INLA

- Gaussian Markov random fields
- Latent Gaussian models
- Laplace approximations

which together (and ++++...) gives a very very nice tool for Bayesian inference

- quick
- accurate (relative error)
- good scaling properties
- +++

Summary

Three main ingredients in INLA

- Gaussian Markov random fields
- Latent Gaussian models
- Laplace approximations

which together (and ++++...) gives a very very nice tool for Bayesian inference

- quick
- accurate (relative error)
- good scaling properties
- +++

Summary

Three main ingredients in INLA

- Gaussian Markov random fields
- Latent Gaussian models
- Laplace approximations

which together (and ++++...) gives a very very nice tool for Bayesian inference

- quick
- accurate (relative error)
- good scaling properties
- +++

Summary

Three main ingredients in INLA

- Gaussian Markov random fields
- Latent Gaussian models
- Laplace approximations

which together (and ++++...) gives a very very nice tool for Bayesian inference

- quick
- accurate (relative error)
- good scaling properties
- +++

Summary

Three main ingredients in INLA

- Gaussian Markov random fields
- Latent Gaussian models
- Laplace approximations

which together (and ++++...) gives a very very nice tool for Bayesian inference

- quick
- accurate (relative error)
- good scaling properties
- +++

Summary

Three main ingredients in INLA

- Gaussian Markov random fields
- Latent Gaussian models
- Laplace approximations

which together (and ++++...) gives a very very nice tool for Bayesian inference

- quick
- accurate (relative error)
- good scaling properties
- +++

Summary

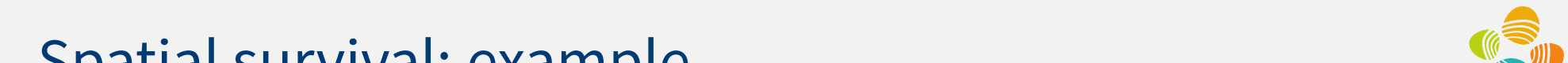
Three main ingredients in INLA

- Gaussian Markov random fields
- Latent Gaussian models
- Laplace approximations

which together (and ++++...) gives a very very nice tool for Bayesian inference

- quick
- accurate (relative error)
- good scaling properties
- +++

Spatial survival: example



Leukaemia survival data (Henderson et al, 2002, JASA), 1043 cases.

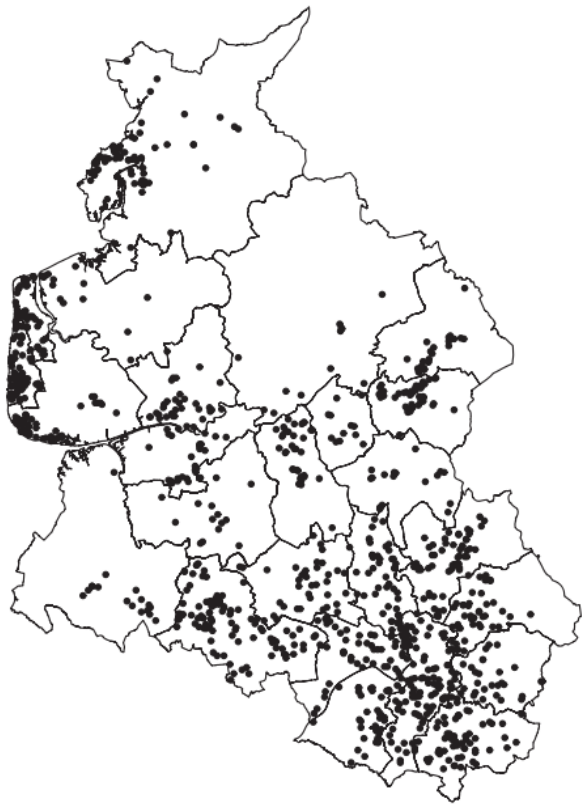


Fig. 1. Leukaemia survival data: districts of Northwest England and locations of the observations.

Spatial survival: example

$$\log(\text{hazard}) = \log(\text{baseline})$$
$$+ f(\text{white blood cell count})$$
$$+ f(\text{deprivation index})$$
$$+ f(\text{spatial})$$
$$+ \text{sex}$$
$$+ \text{age}$$

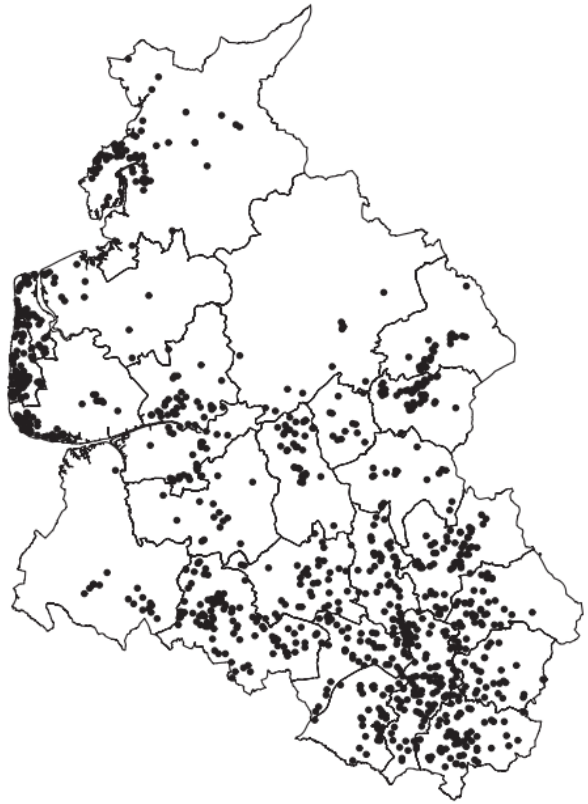
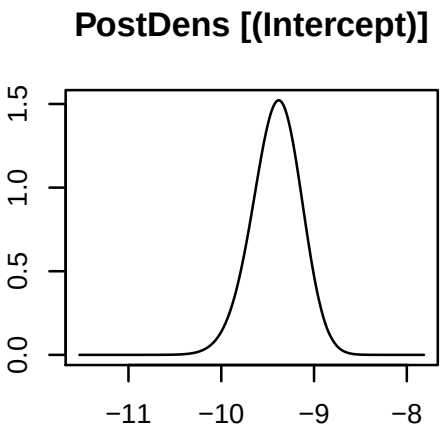


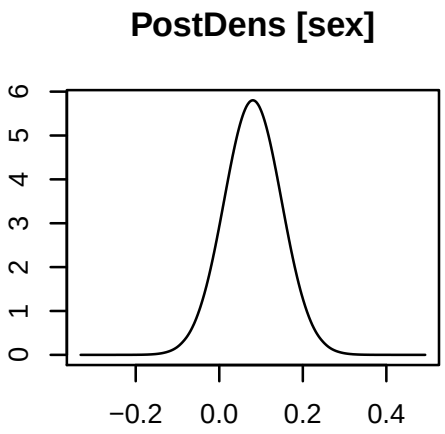
Fig. 1. Leukaemia survival data: districts of Northwest England and locations of the obse

R-code

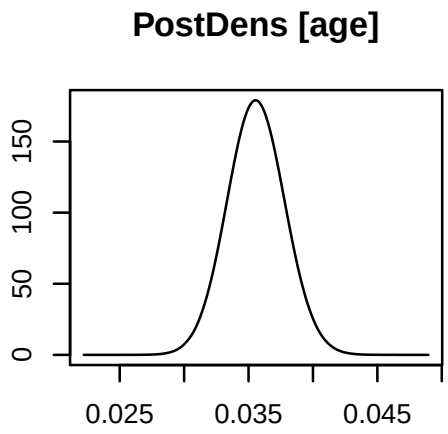
```
data(Leuk)
g = system.file("demodata/Leuk.graph", package="INLA")

formula = inla.surv(Leuk$time, Leuk$cens) ~ sex + age +
    f(inla.group(wbc), model="rw2") +
    f(inla.group(tpi), model="rw2") +
    f(district, model="besag", graph = g) +

r = inla(formula, family="coxph", data = Leuk)

plot(r)
```



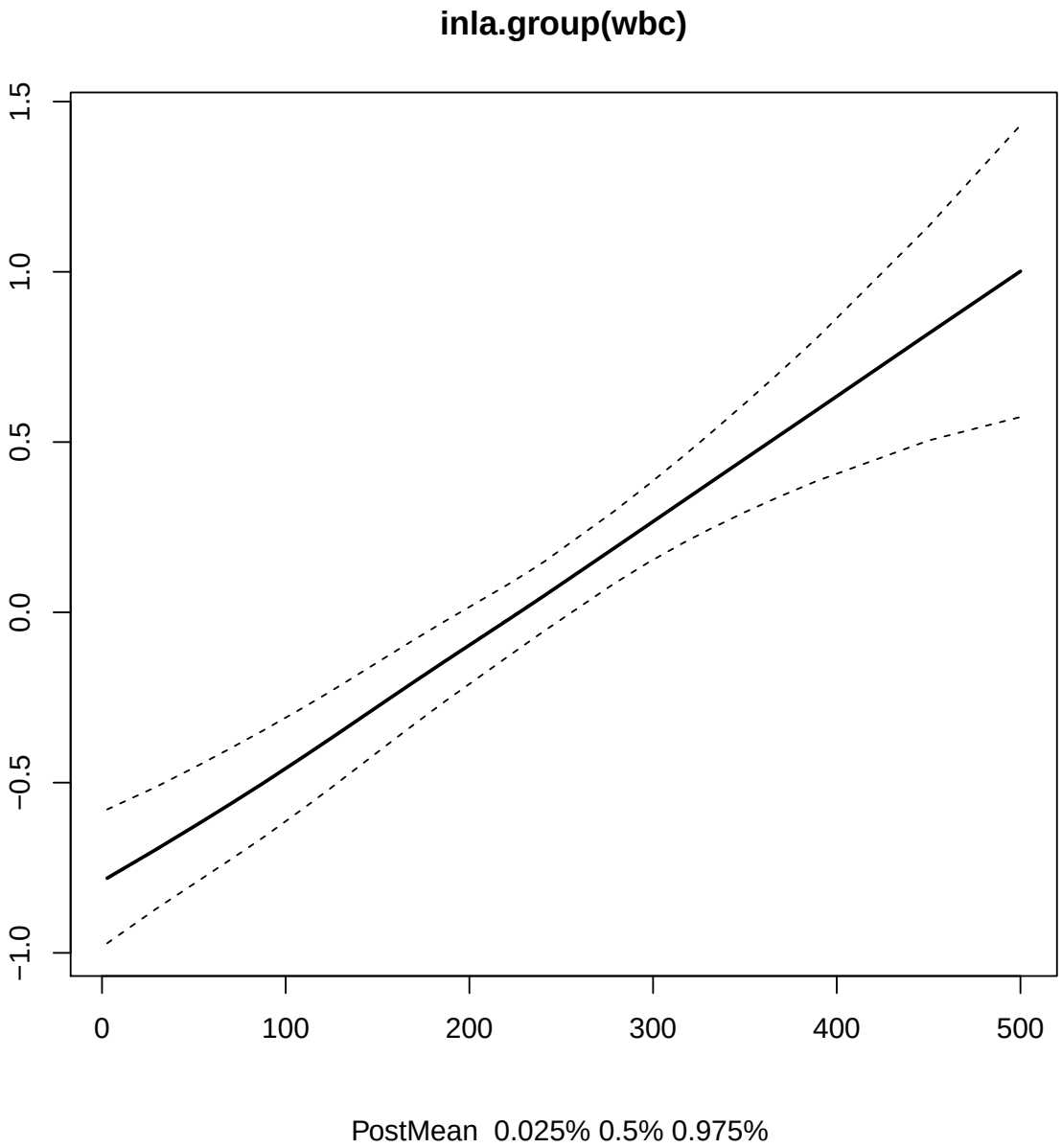
Mean = −9.407 SD = 0.265

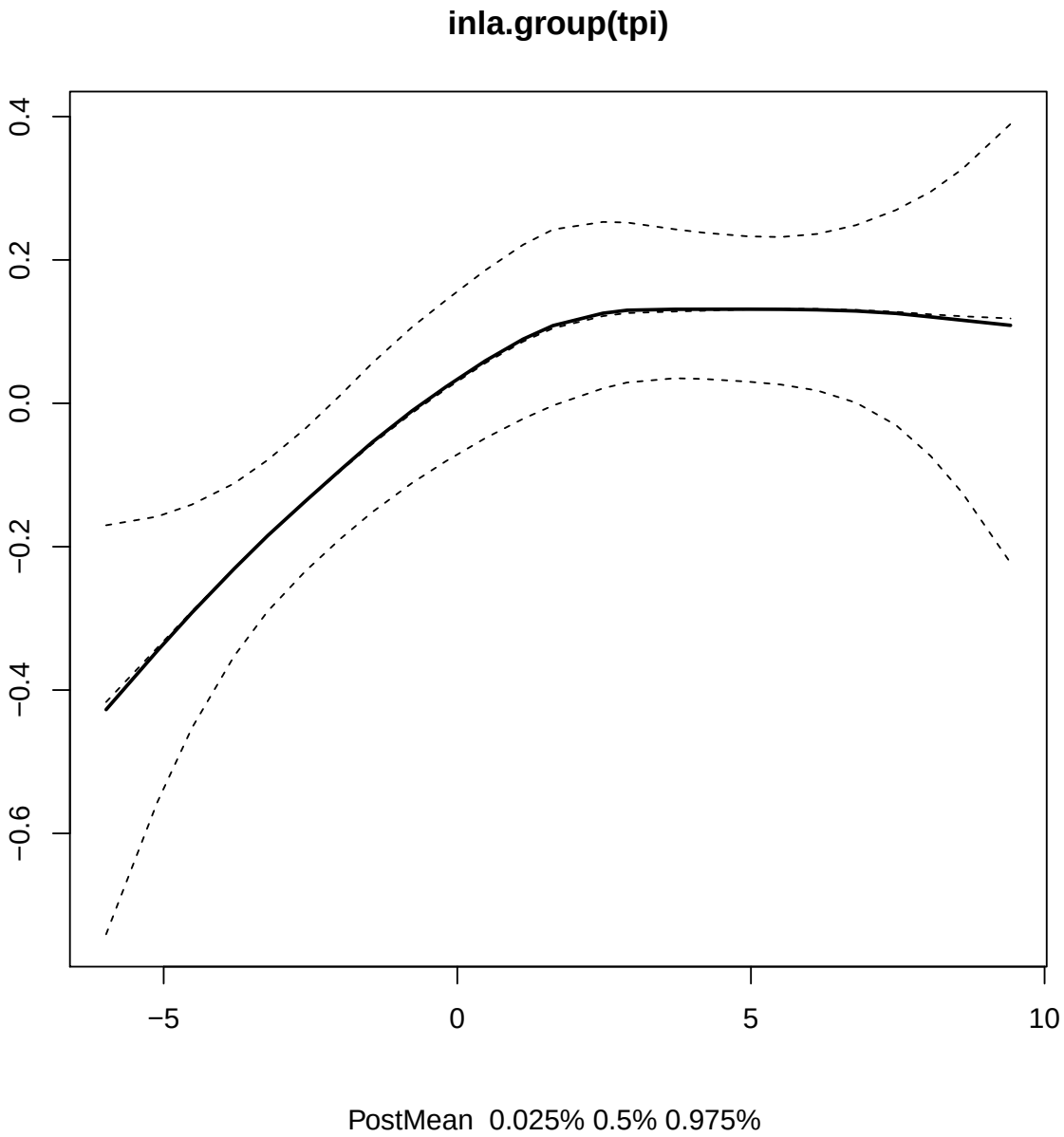


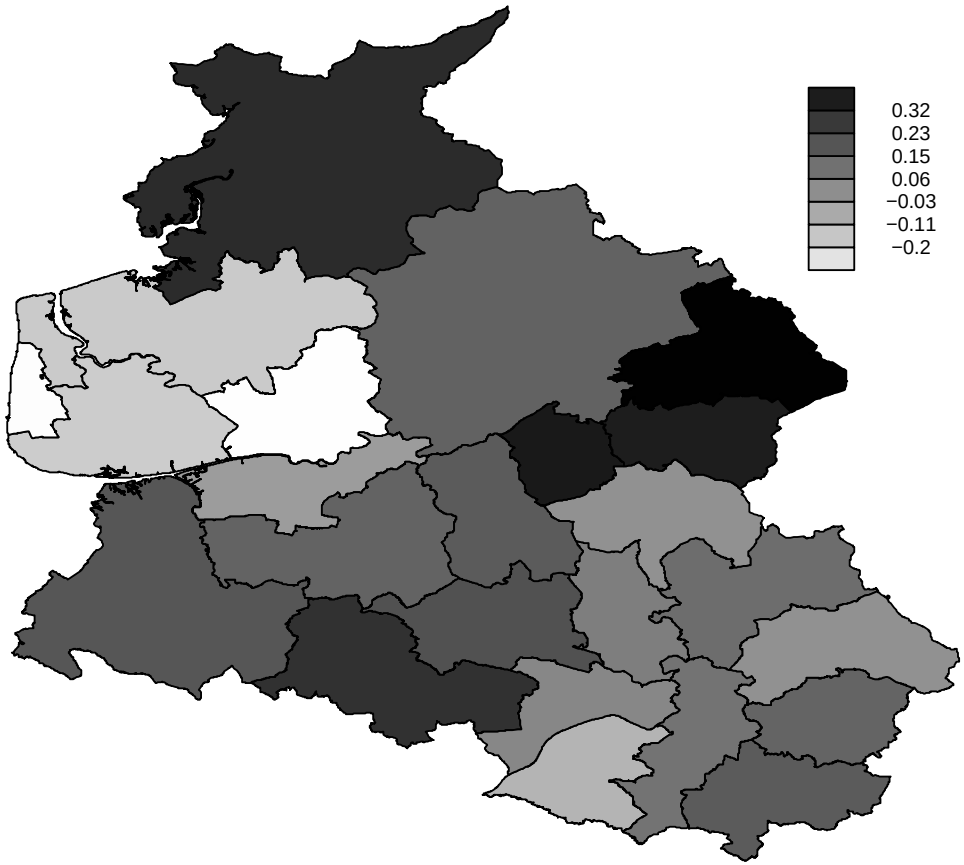
Mean = 0.08 SD = 0.069

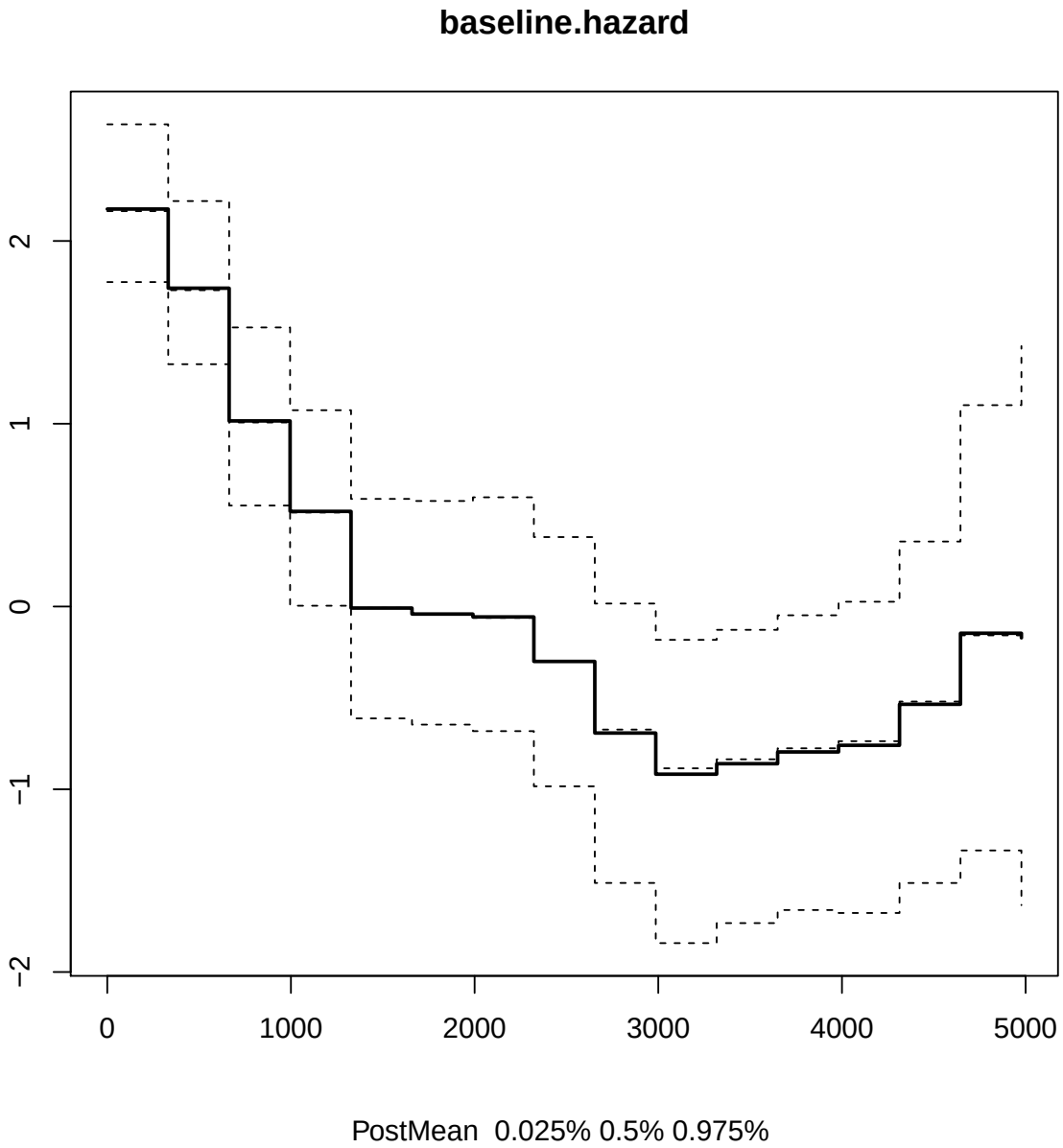


Mean = 0.036 SD = 0.002


inla.group(wbc)
1.5
1.0
0.5
0.0
−0.5
−1.0
0
100
200
300
400
500
PostMean 0.025% 0.5% 0.975%


inla.group(tpi)
0.4
0.2
0.0
−0.2
−0.4
−0.6
−5
0
5
10
PostMean 0.025% 0.5% 0.975%


0.32
0.23
0.15
0.06
−0.03
−0.11
−0.2


baseline.hazard
2
1
0
−1
−2
0
1000
2000
3000
4000
5000
PostMean 0.025% 0.5% 0.975%

Some internal statistics

Running time on my laptop: 2.3 seconds

- Factorise $\boldsymbol{Q}$ (dim $= 2453$): 455 times
- Solve $\boldsymbol{Qx} = \boldsymbol{b}$: 3160 times
- Partially invert $\boldsymbol{Q}$: 28 times

Part II

Theory and methods

What is a Gaussian Markov random field (GMRF)?

A GMRF is a simple construct

- A normal distributed random vector

$$\boldsymbol{x} = (x_1, \ldots, x_n)^T$$

- Additional Markov properties:

$$x_i \perp x_j \mid \boldsymbol{x}_{-ij}$$

x_i and x_j are conditional independent (CI).

What is a Gaussian Markov random field (GMRF)?

A GMRF is a simple construct

- A normal distributed random vector

$$\boldsymbol{x} = (x_1, \ldots, x_n)^T$$

- Additional Markov properties:

$$x_i \perp x_j \mid \boldsymbol{x}_{-ij}$$

x_i and x_j are conditional independent (CI).

If $x_i \perp x_j \mid \boldsymbol{x}_{-ij}$ for a set of $\{i, j\}$, then we need to constrain the parametrisation of the GMRF.

- Covariance matrix: difficult
- Precision matrix: easy

If $x_i \perp x_j \mid \boldsymbol{x}_{-ij}$ for a set of $\{i, j\}$, then we need to constrain the parametrisation of the GMRF.

- Covariance matrix: difficult
- Precision matrix: easy

Conditional independence and the precision matrix

The density of a zero mean Gaussian

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

Constraining the parametrisation to obey CI properties

Theorem

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

Conditional independence and the precision matrix

The density of a zero mean Gaussian

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

Constraining the parametrisation to obey CI properties

Theorem

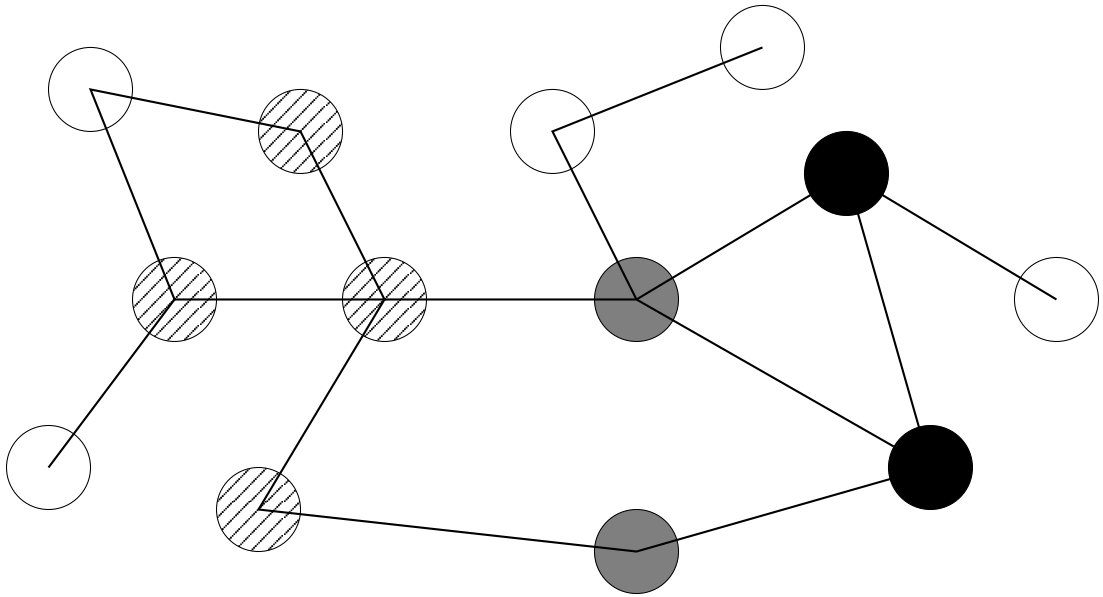
$$x_i \perp x_j \mid \boldsymbol{x}_{-ij} \quad \Longleftrightarrow \quad Q_{ij} = 0$$

Global Markov property

Let $\boldsymbol{x}$ be a GMRF wrt $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.
The global Markov property:

$$\boldsymbol{x}_A \perp \boldsymbol{x}_B \mid \boldsymbol{x}_C$$

for all disjoint sets A, B and C where C separates A and B, and A and B are non-empty.

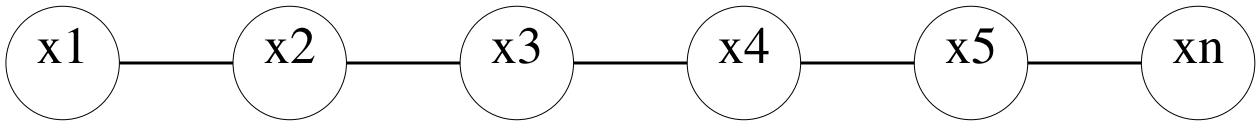


Simple example of a GMRF

Auto-regressive process of order 1

$$x_t \mid x_{t-1}, \ldots, x_1 \sim \mathcal{N}(\phi x_{t-1}, 1), \quad t = 2, \ldots, n$$

and $x_1 \sim \mathcal{N}(0, (1-\phi^2)^{-1})$.



Tridiagonal precision matrix

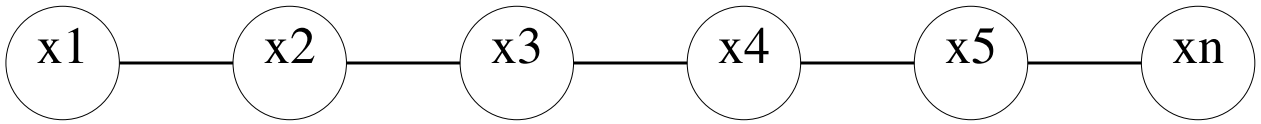
$$Q = \begin{pmatrix} 1 & -\phi & & & & \\ -\phi & 1+\phi^2 & -\phi & & & \\ & \ddots & \ddots & \ddots & & \\ & & -\phi & 1+\phi^2 & -\phi & \\ & & & -\phi & 1 & \end{pmatrix}$$

Simple example of a GMRF

Auto-regressive process of order 1

$$x_t \mid x_{t-1}, \ldots, x_1 \sim \mathcal{N}(\phi x_{t-1}, 1), \quad t = 2, \ldots, n$$

and $x_1 \sim \mathcal{N}(0, (1-\phi^2)^{-1})$.



Tridiagonal precision matrix

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

Part III

Simulation algorithms for GMRFs. Numerical methods for sparse matrices.

Cholesky factorisation

- If $\boldsymbol{A} > 0$ be a $n \times n$ positive definite matrix, then there exists a unique Cholesky triangle $\boldsymbol{L}$, such that $\boldsymbol{L}$ is a lower triangular matrix, and

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^T$$

- Computing $\boldsymbol{L}$ costs $n^3/3$ flops.
- This factorisation is the basis for *solving* systems like

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \quad \text{or} \quad \boldsymbol{A}\boldsymbol{X} = \boldsymbol{B}$$

for k right hand sides, or equivalently, computing

$$\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b} \quad \text{or} \quad \boldsymbol{X} = \boldsymbol{A}^{-1}\boldsymbol{B}$$

Cholesky factorisation

- If $\boldsymbol{A} > 0$ be a $n \times n$ positive definite matrix, then there exists a unique Cholesky triangle $\boldsymbol{L}$, such that $\boldsymbol{L}$ is a lower triangular matrix, and

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^T$$

- Computing $\boldsymbol{L}$ costs $n^3/3$ flops.
- This factorisation is the basis for *solving* systems like

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \quad \text{or} \quad \boldsymbol{A}\boldsymbol{X} = \boldsymbol{B}$$

for k right hand sides, or equivalently, computing

$$\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b} \quad \text{or} \quad \boldsymbol{X} = \boldsymbol{A}^{-1}\boldsymbol{B}$$

Cholesky factorisation

- If $\boldsymbol{A} > 0$ be a $n \times n$ positive definite matrix, then there exists a unique Cholesky triangle $\boldsymbol{L}$, such that $\boldsymbol{L}$ is a lower triangular matrix, and

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^T$$

- Computing $\boldsymbol{L}$ costs $n^3/3$ flops.
- This factorisation is the basis for *solving* systems like

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \quad \text{or} \quad \boldsymbol{A}\boldsymbol{X} = \boldsymbol{B}$$

for k right hand sides, or equivalently, computing

$$\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b} \quad \text{or} \quad \boldsymbol{X} = \boldsymbol{A}^{-1}\boldsymbol{B}$$

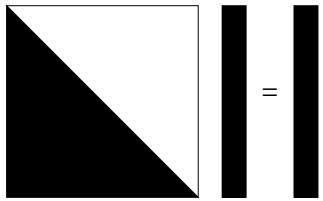
Algorithm 1 Solving $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ where $\boldsymbol{A} > 0$

1: Compute the Cholesky factorisation, $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^T$
2: Solve $\boldsymbol{L}\boldsymbol{v} = \boldsymbol{b}$
3: Solve $\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{v}$
4: **Return** $\boldsymbol{x}$

Algorithm 1 Solving $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ where $\boldsymbol{A} > 0$

1: Compute the Cholesky factorisation, $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^T$
2: Solve $\boldsymbol{L}\boldsymbol{v} = \boldsymbol{b}$
3: Solve $\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{v}$
4: **Return** $\boldsymbol{x}$

Step 2 is called *forward-substitution* and cost $\mathcal{O}(n^2)$ flops.



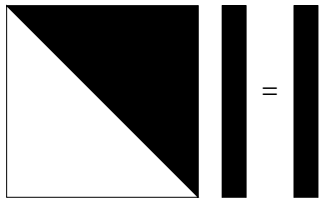
The solution $\boldsymbol{v}$ is computed in a forward-loop

$$v_i = \frac{1}{L_{ii}}(b_i - \sum_{j=1}^{i-1} L_{ij}v_j), \quad i = 1, \ldots, n \tag{1}$$

Algorithm 1 Solving $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ where $\boldsymbol{A} > 0$

1: Compute the Cholesky factorisation, $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^T$
2: Solve $\boldsymbol{L}\boldsymbol{v} = \boldsymbol{b}$
3: Solve $\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{v}$
4: **Return** $\boldsymbol{x}$

Step 3 is called *back-substitution* and costs $\mathcal{O}(n^2)$ flops.



The solution $\boldsymbol{x}$ is computed in a backward-loop

$$x_i = \frac{1}{L_{ii}}(v_i - \sum_{j=i+1}^{n} L_{ji}x_j), \quad i = n, \ldots, 1 \tag{2}$$

To compute $\boldsymbol{A}^{-1}\boldsymbol{B}$ where $\boldsymbol{B}$ is a $n \times k$ matrix, we do this by computing the solution $\boldsymbol{X}$ of

$$\boldsymbol{A}\boldsymbol{X}_j = \boldsymbol{B}_j$$

for each of the k columns of $\boldsymbol{X}$.

Algorithm 2 Solving $\boldsymbol{A}\boldsymbol{X} = \boldsymbol{B}$ where $\boldsymbol{A} > 0$

1: Compute the Cholesky factorisation, $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^T$
2: **for** $j = 1$ to k **do**
3: Solve $\boldsymbol{L}\boldsymbol{v} = \boldsymbol{B}_j$
4: Solve $\boldsymbol{L}^T\boldsymbol{X}_j = \boldsymbol{v}$
5: **end for**
6: **Return** $\boldsymbol{X}$

To compute $\boldsymbol{A}^{-1}\boldsymbol{B}$ where $\boldsymbol{B}$ is a $n \times k$ matrix, we do this by computing the solution $\boldsymbol{X}$ of

$$\boldsymbol{A}\boldsymbol{X}_j = \boldsymbol{B}_j$$

for each of the k columns of $\boldsymbol{X}$.

Algorithm 2 Solving $\boldsymbol{A}\boldsymbol{X} = \boldsymbol{B}$ where $\boldsymbol{A} > 0$

1: Compute the Cholesky factorisation, $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^T$
2: **for** $j = 1$ to k **do**
3: Solve $\boldsymbol{L}\boldsymbol{v} = \boldsymbol{B}_j$
4: Solve $\boldsymbol{L}^T\boldsymbol{X}_j = \boldsymbol{v}$
5: **end for**
6: **Return** $\boldsymbol{X}$

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

If $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$ and $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, then $\boldsymbol{x}$ defined by

$$\boldsymbol{L}^T \boldsymbol{x} = \boldsymbol{z}$$

has covariance

$$\text{Cov}(\boldsymbol{x}) = \text{Cov}(\boldsymbol{L}^{-T}\boldsymbol{z}) = (\boldsymbol{L}\boldsymbol{L}^T)^{-1} = \boldsymbol{Q}^{-1}$$

Algorithm 3 Sampling $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{Q}^{-1})$

1: Compute the Cholesky factorisation, $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$
2: Sample $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$
3: Solve $\boldsymbol{L}^T \boldsymbol{v} = \boldsymbol{z}$
4: Compute $\boldsymbol{x} = \boldsymbol{\mu} + \boldsymbol{v}$
5: **Return** $\boldsymbol{x}$

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

If $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$ and $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, then $\boldsymbol{x}$ defined by

$$\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{z}$$

has covariance

$$\text{Cov}(\boldsymbol{x}) = \text{Cov}(\boldsymbol{L}^{-T}\boldsymbol{z}) = (\boldsymbol{L}\boldsymbol{L}^T)^{-1} = \boldsymbol{Q}^{-1}$$

Algorithm 3 Sampling $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{Q}^{-1})$

1: Compute the Cholesky factorisation, $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$
2: Sample $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$
3: Solve $\boldsymbol{L}^T\boldsymbol{v} = \boldsymbol{z}$
4: Compute $\boldsymbol{x} = \boldsymbol{\mu} + \boldsymbol{v}$
5: **Return** $\boldsymbol{x}$

The log-density is

$$\log \pi(\boldsymbol{x}) = -\frac{n}{2}\log 2\pi + \sum_{i=1}^{n} \log L_{ii} - \frac{1}{2}\underbrace{(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{Q}(\boldsymbol{x}-\boldsymbol{\mu})}_{=q}$$

If $\boldsymbol{x}$ is sampled, then

$$q = \boldsymbol{z}^T\boldsymbol{z}$$

otherwise, compute this term as

- $\boldsymbol{u} = \boldsymbol{x} - \mu$
- $\boldsymbol{v} = \boldsymbol{Q}\boldsymbol{u}$
- $q = \boldsymbol{u}^T\boldsymbol{v}$

The log-density is

$$\log \pi(\boldsymbol{x}) = -\frac{n}{2}\log 2\pi + \sum_{i=1}^{n} \log L_{ii} - \frac{1}{2}\underbrace{(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{Q}(\boldsymbol{x}-\boldsymbol{\mu})}_{=q}$$

If $\boldsymbol{x}$ is sampled, then

$$q = \boldsymbol{z}^T\boldsymbol{z}$$

otherwise, compute this term as

- $\boldsymbol{u} = \boldsymbol{x} - \mu$
- $\boldsymbol{v} = \boldsymbol{Q}\boldsymbol{u}$
- $q = \boldsymbol{u}^T\boldsymbol{v}$

The log-density is

$$\log \pi(\boldsymbol{x}) = -\frac{n}{2}\log 2\pi + \sum_{i=1}^{n} \log L_{ii} - \frac{1}{2}\underbrace{(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{Q}(\boldsymbol{x}-\boldsymbol{\mu})}_{=q}$$

If $\boldsymbol{x}$ is sampled, then

$$q = \boldsymbol{z}^T\boldsymbol{z}$$

otherwise, compute this term as

- $\boldsymbol{u} = \boldsymbol{x} - \boldsymbol{\mu}$
- $\boldsymbol{v} = \boldsymbol{Q}\boldsymbol{u}$
- $q = \boldsymbol{u}^T\boldsymbol{v}$

Part IV

Numerical methods for sparse matrices

Numerical methods for sparse matrices

Computations on GMRFs can be expressed such that the main tasks are

1. compute the Cholesky factorisation of $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$, and
2. solve $\boldsymbol{L}\boldsymbol{v} = \boldsymbol{b}$ and $\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{z}$.
3. The second task is much-faster than the first, but sparsity will be of advantage also here.

Numerical methods for sparse matrices

Computations on GMRFs can be expressed such that the main tasks are

1. compute the Cholesky factorisation of $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$, and
2. solve $\boldsymbol{L}\boldsymbol{v} = \boldsymbol{b}$ and $\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{z}$.
3. The second task is much-faster than the first, but sparsity will be of advantage also here.

Numerical methods for sparse matrices

Computations on GMRFs can be expressed such that the main tasks are

1. compute the Cholesky factorisation of $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$, and
2. solve $\boldsymbol{L}\boldsymbol{v} = \boldsymbol{b}$ and $\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{z}$.
3. The second task is much-faster than the first, but sparsity will be of advantage also here.

The questions are

- **why** a sparse $\boldsymbol{Q}$ allow for fast factorisation,
- **how** we can take advantage of it,
- **why** we gain if we permute the vertics before factorising the matrix

The questions are

- **why** a sparse $\boldsymbol{Q}$ allow for fast factorisation,
- **how** we can take advantage of it,
- **why** we gain if we permute the vertics before factorising the matrix

The questions are

- **why** a sparse $\boldsymbol{Q}$ allow for fast factorisation,
- **how** we can take advantage of it,
- **why** we gain if we permute the vertics before factorising the matrix

How to compute the Cholesky factorisation

$$\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$$

$$Q_{ij} = \sum_{k=1}^{j} L_{ik}L_{jk}, \qquad v_i = Q_{ij} - \sum_{k=1}^{j-1} L_{ik}L_{jk}, \quad i \geq j,$$

Then

- $L_{jj}^2 = v_j$, and
- $L_{ij}L_{jj} = v_i$ for $i > j$.

If we know $\{v_i\}$ for fixed j, then

$$L_{jj} = \sqrt{v_j} \quad \text{and} \quad L_{ij} = v_i/\sqrt{v_j}, \quad \text{for} \quad i = j+1, \ldots, n.$$

This gives the jth column in $\boldsymbol{L}$.

How to compute the Cholesky factorisation

$$\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$$

$$Q_{ij} = \sum_{k=1}^{j} L_{ik}L_{jk}, \qquad v_i = Q_{ij} - \sum_{k=1}^{j-1} L_{ik}L_{jk}, \quad i \geq j,$$

Then

- $L_{jj}^2 = v_j$, and
- $L_{ij}L_{jj} = v_i$ for $i > j$.

If we know $\{v_i\}$ for fixed j, then

$$L_{jj} = \sqrt{v_j} \quad \text{and} \quad L_{ij} = v_i/\sqrt{v_j}, \quad \text{for} \quad i = j+1, \ldots, n.$$

This gives the jth column in $\boldsymbol{L}$.

How to compute the Cholesky factorisation

$$\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$$

$$Q_{ij} = \sum_{k=1}^{j} L_{ik}L_{jk}, \qquad v_i = Q_{ij} - \sum_{k=1}^{j-1} L_{ik}L_{jk}, \quad i \geq j,$$

Then

- $L_{jj}^2 = v_j$, and
- $L_{ij}L_{jj} = v_i$ for $i > j$.

If we know $\{v_i\}$ for fixed j, then

$$L_{jj} = \sqrt{v_j} \quad \text{and} \quad L_{ij} = v_i/\sqrt{v_j}, \quad \text{for} \quad i = j+1, \ldots, n.$$

This gives the jth column in $\boldsymbol{L}$.

Cholesky factorization of $\boldsymbol{Q} > 0$

Algorithm 4 Computing the Cholesky triangle $\boldsymbol{L}$ of $\boldsymbol{Q}$

1: **for** $j = 1$ **to** n **do**
2: $\quad v_{j:n} = Q_{j:n,j}$
3: $\quad$ **for** $k = 1$ **to** $j-1$ **do** $v_{j:n} = v_{j:n} - L_{j:n,k} L_{jk}$
4: $\quad L_{j:n,j} = v_{j:n} / \sqrt{v_j}$
5: **end for**
6: **Return** $\boldsymbol{L}$

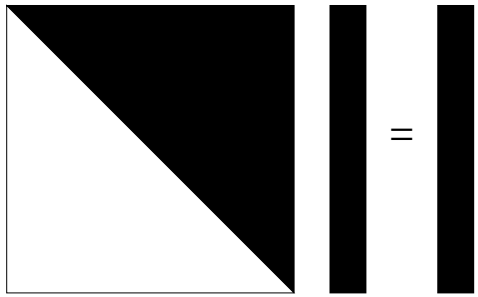
The overall process involves $n^3/3$ flops.

Interpretation of $\boldsymbol{L}$ (I)

Let $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$, then the solution of

$$\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{z} \quad \text{where} \quad \boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$$

is $\mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}^{-1})$ distributed.



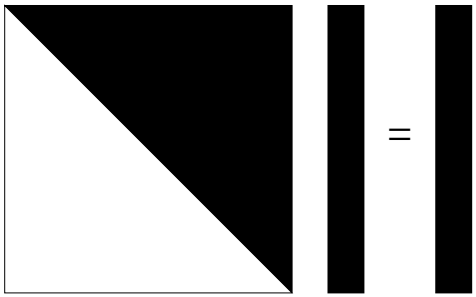
Since $\boldsymbol{L}$ is lower triangular then

$$x_n = \frac{1}{L_{nn}} z_n$$

$$x_{n-1} = \frac{1}{L_{n-1,n-1}} \left(z_{n-1} - L_{n,n-1} x_n\right)$$

$$\dots$$

Interpretation of $\boldsymbol{L}$ (I)



Let $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$, then the solution of

$$\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{z} \quad \text{where} \quad \boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$$

is $\mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}^{-1})$ distributed.

Since $\boldsymbol{L}$ is lower triangular then

$$x_n = \frac{1}{L_{nn}} z_n$$

$$x_{n-1} = \frac{1}{L_{n-1,n-1}} \left(z_{n-1} - L_{n,n-1} x_n\right)$$

$$\dots$$

Interpretation of $\boldsymbol{L}$ (II)

Theorem

Let $\boldsymbol{x}$ be a GMRF wrt to the labelled graph $\mathcal{G}$, with mean $\mathbf{0}$ and precision matrix $\boldsymbol{Q} > 0$. Let $\boldsymbol{L}$ be the Cholesky triangle of $\boldsymbol{Q}$. Then for $i \in \mathcal{V}$,

$$E(x_i \mid \boldsymbol{x}_{(i+1):n}) = -\frac{1}{L_{ii}} \sum_{j=i+1}^{n} L_{ji} x_j \qquad \text{and}$$

$$Prec(x_i \mid \boldsymbol{x}_{(i+1):n}) = L_{ii}^2.$$

Determine the zero-pattern in $\boldsymbol{L}$ (I)

Theorem

Let $\boldsymbol{x}$ be a GMRF wrt $\mathcal{G}$, with mean $\mathbf{0}$ and precision matrix $\boldsymbol{Q} > 0$. Let $\boldsymbol{L}$ be the Cholesky triangle of $\boldsymbol{Q}$ and define for $1 \le i < j \le n$ the set

$$F(i,j) = \{i+1, \ldots, j-1, j+1, \ldots, n\},$$

which is the future of i except j. Then

$$x_i \perp x_j \mid \boldsymbol{x}_{F(i,j)} \quad \Longleftrightarrow \quad L_{ji} = 0.$$

If we can verify that L_{ji} is zero, we do not have to compute it when factorising $\boldsymbol{Q}$

Determine the zero-pattern in $\boldsymbol{L}$ (I)

Theorem

Let $\boldsymbol{x}$ be a GMRF wrt $\mathcal{G}$, with mean $\boldsymbol{0}$ and precision matrix $\boldsymbol{Q} > 0$. Let $\boldsymbol{L}$ be the Cholesky triangle of $\boldsymbol{Q}$ and define for $1 \le i < j \le n$ the set

$$F(i,j) = \{i+1, \ldots, j-1, j+1, \ldots, n\},$$

which is the future of i except j. Then

$$x_i \perp x_j \mid \boldsymbol{x}_{F(i,j)} \quad \Longleftrightarrow \quad L_{ji} = 0.$$

If we can verify that L_{ji} is zero, we do not have to compute it when factorising $\boldsymbol{Q}$

Proof

Assume $\boldsymbol{\mu} = \mathbf{0}$ and fix $1 \le i < j \le n$. Theorem 3 gives that

$$\begin{aligned} \pi(\boldsymbol{x}_{i:n}) &\propto \exp\left(-\frac{1}{2}\sum_{k=i}^{n} L_{kk}^2 \left(x_k + \frac{1}{L_{kk}}\sum_{j=k+1}^{n} L_{jk}x_j\right)^2\right) \\ &= \exp\left(-\frac{1}{2}\boldsymbol{x}_{i:n}^T \boldsymbol{Q}^{(i:n)}\boldsymbol{x}_{i:n}\right), \end{aligned}$$

where $Q_{ij}^{(i:n)} = L_{ii}L_{ji}$. Then

$$x_i \perp x_j \mid \boldsymbol{x}_{F(i,j)} \quad \Longleftrightarrow \quad L_{ii}L_{ji} = 0,$$

which is equivalent to $L_{ji} = 0$ since $L_{ii} > 0$ as $\boldsymbol{Q}^{(i:n)} > 0$.

Proof

Assume $\boldsymbol{\mu} = \mathbf{0}$ and fix $1 \le i < j \le n$. Theorem 3 gives that

$$\begin{aligned} \pi(\boldsymbol{x}_{i:n}) &\propto \exp\left(-\frac{1}{2}\sum_{k=i}^{n} L_{kk}^2 \left(x_k + \frac{1}{L_{kk}}\sum_{j=k+1}^{n} L_{jk}x_j\right)^2\right) \\ &= \exp\left(-\frac{1}{2}\boldsymbol{x}_{i:n}^T \boldsymbol{Q}^{(i:n)}\boldsymbol{x}_{i:n}\right), \end{aligned}$$

where $Q_{ij}^{(i:n)} = L_{ii}L_{ji}$. Then

$$x_i \perp x_j \mid \boldsymbol{x}_{F(i,j)} \quad \Longleftrightarrow \quad L_{ii}L_{ji} = 0,$$

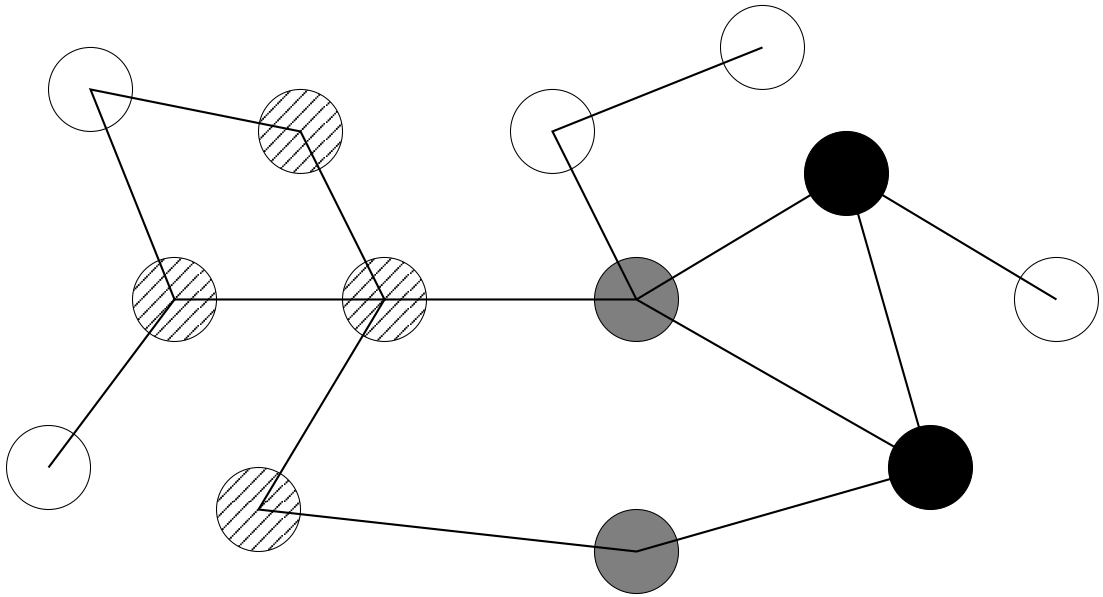
which is equivalent to $L_{ji} = 0$ since $L_{ii} > 0$ as $\boldsymbol{Q}^{(i:n)} > 0$.

(Global Markov property)

Let $\boldsymbol{x}$ be a GMRF wrt $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.
The global Markov property:

$$\boldsymbol{x}_A \perp \boldsymbol{x}_B \mid \boldsymbol{x}_C$$

for all disjoint sets A, B and C where C separates A and B, and A and B are non-empty.



Determine the zero-pattern in $\boldsymbol{L}$ (II)

The **global Markov property** provide a simple and sufficient criteria for checking if $L_{ji} = 0$.

Corollary

If $F(i,j)$ separates $i < j$ in $\mathcal{G}$, then $L_{ji} = 0$.

Corollary

If $i \sim j$ then $F(i,j)$ does not separates $i < j$.

The idea is simple

- Use the *global Markov property* to check if $L_{ji} = 0$.
- Compute only the non-zero terms in $\boldsymbol{L}$, so that $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$.

Determine the zero-pattern in $\boldsymbol{L}$ (II)

The **global Markov property** provide a simple and sufficient criteria for checking if $L_{ji} = 0$.

Corollary

If $F(i, j)$ separates $i < j$ in $\mathcal{G}$, then $L_{ji} = 0$.

Corollary

If $i \sim j$ then $F(i, j)$ does not separates $i < j$.

The idea is simple

- Use the *global Markov property* to check if $L_{ji} = 0$.
- Compute only the non-zero terms in $\boldsymbol{L}$, so that $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$.

Determine the zero-pattern in $\boldsymbol{L}$ (II)

The **global Markov property** provide a simple and sufficient criteria for checking if $L_{ji} = 0$.

Corollary

If $F(i,j)$ separates $i < j$ in $\mathcal{G}$, then $L_{ji} = 0$.

Corollary

If $i \sim j$ then $F(i,j)$ does not separates $i < j$.

The idea is simple

- Use the *global Markov property* to check if $L_{ji} = 0$.
- Compute only the non-zero terms in $\boldsymbol{L}$, so that $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$.

Determine the zero-pattern in $\boldsymbol{L}$ (II)

The **global Markov property** provide a simple and sufficient criteria for checking if $L_{ji} = 0$.

Corollary

If $F(i,j)$ separates $i < j$ in $\mathcal{G}$, then $L_{ji} = 0$.

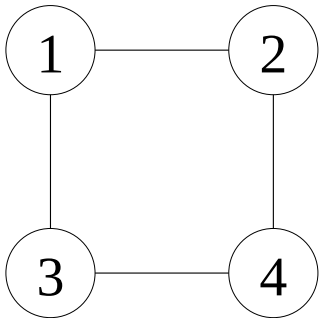
Corollary

If $i \sim j$ then $F(i,j)$ does not separates $i < j$.

The idea is simple

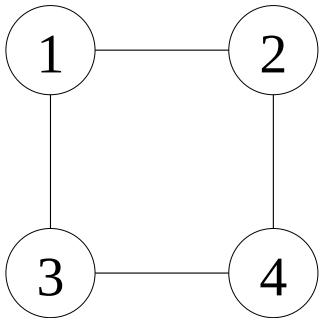
- Use the *global Markov property* to check if $L_{ji} = 0$.
- Compute only the non-zero terms in $\boldsymbol{L}$, so that $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$.

Example



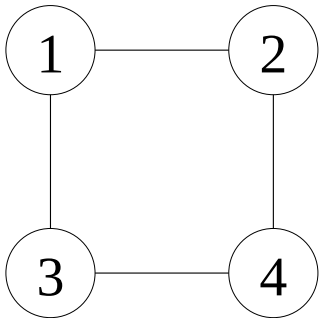
$$Q = \begin{pmatrix} \times & \times & \times & \\ \times & \times & & \times \\ \times & & \times & \times \\ & \times & \times & \times \end{pmatrix}$$

Example



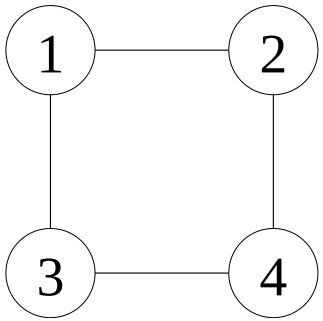
$$\boldsymbol{Q} = \begin{pmatrix} \times & \times & \times & \\ \times & \times & & \times \\ \times & & \times & \times \\ & \times & \times & \times \end{pmatrix}$$

Example



$$\boldsymbol{Q} = \begin{pmatrix} \times & \times & \times & \\ \times & \times & & \times \\ \times & & \times & \times \\ & \times & \times & \times \end{pmatrix} \qquad \boldsymbol{L} = \begin{pmatrix} \times & & & \\ \times & \times & & \\ \times & ? & \times & \\ ? & \times & \times & \times \end{pmatrix}$$

Example



$$\boldsymbol{Q} = \begin{pmatrix} \times & \times & \times & \\ \times & \times & & \times \\ \times & & \times & \times \\ & \times & \times & \times \end{pmatrix} \qquad \boldsymbol{L} = \begin{pmatrix} \times & & & \\ \times & \times & & \\ \times & \surd & \times & \\ & \times & \times & \times \end{pmatrix}$$

Example: AR(1)-process

$$x_t \mid \boldsymbol{x}_{1:(t-1)} \sim \mathcal{N}(\phi x_{t-1}, \sigma^2), \quad t = 1, \ldots, n$$

$$Q = \begin{pmatrix} \times & \times & & & & & \\ \times & \times & \times & & & & \\ & \times & \times & \times & & & \\ & & \times & \times & \times & & \\ & & & \times & \times & \times & \\ & & & & \times & \times & \times \\ & & & & & \times & \times \end{pmatrix} \qquad L = \begin{pmatrix} \times & & & & & & \\ \times & \times & & & & & \\ & \times & \times & & & & \\ & & \times & \times & & & \\ & & & \times & \times & & \\ & & & & \times & \times & \\ & & & & & \times & \times \end{pmatrix}$$

Example: AR(1)-process

$$x_t \mid \boldsymbol{x}_{1:(t-1)} \sim \mathcal{N}(\phi x_{t-1}, \sigma^2), \quad t = 1, \ldots, n$$

$$\boldsymbol{Q} = \begin{pmatrix} \times & \times & & & & & \\ \times & \times & \times & & & & \\ & \times & \times & \times & & & \\ & & \times & \times & \times & & \\ & & & \times & \times & \times & \\ & & & & \times & \times & \times \\ & & & & & \times & \times \end{pmatrix} \qquad \boldsymbol{L} = \begin{pmatrix} \times & & & & & & \\ \times & \times & & & & & \\ & \times & \times & & & & \\ & & \times & \times & & & \\ & & & \times & \times & & \\ & & & & \times & \times & \\ & & & & & \times & \times \end{pmatrix}$$

Example: AR(1)-process

$$x_t \mid \boldsymbol{x}_{1:(t-1)} \sim \mathcal{N}(\phi x_{t-1}, \sigma^2), \quad t = 1, \ldots, n$$

$$\boldsymbol{Q} = \begin{pmatrix} \times & \times & & & & & \\ \times & \times & \times & & & & \\ & \times & \times & \times & & & \\ & & \times & \times & \times & & \\ & & & \times & \times & \times & \\ & & & & \times & \times & \times \\ & & & & & \times & \times \end{pmatrix} \quad \boldsymbol{L} = \begin{pmatrix} \times & & & & & & \\ \times & \times & & & & & \\ & \times & \times & & & & \\ & & \times & \times & & & \\ & & & \times & \times & & \\ & & & & \times & \times & \\ & & & & & \times & \times \end{pmatrix}$$

Bandwidth is preserved

Similarly, for an AR(p)-process

- $\boldsymbol{Q}$ have bandwidth p.
- $\boldsymbol{L}$ have lower-bandwidth p.

Theorem

Let $\boldsymbol{Q} > 0$ be a band matrix with bandwidth p and dimension n, then the Cholesky triangle of $\boldsymbol{Q}$ has (lower) bandwidth p.

...easy to modify existing Cholesky-factorisation code to use only entries where $|i - j| \leq p$.

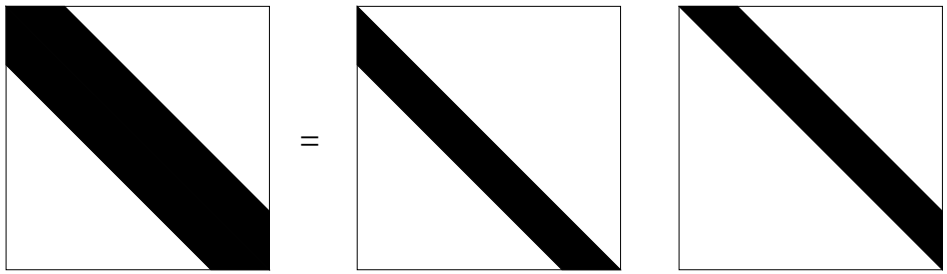
Bandwidth is preserved

Similarly, for an AR(p)-process

- $\boldsymbol{Q}$ have bandwidth p.
- $\boldsymbol{L}$ have lower-bandwidth p.

Theorem

Let $\boldsymbol{Q} > 0$ be a band matrix with bandwidth p and dimension n, then the Cholesky triangle of $\boldsymbol{Q}$ has (lower) bandwidth p.



...easy to modify existing Cholesky-factorisation code to use only entries where $|i - j| \leq p$.

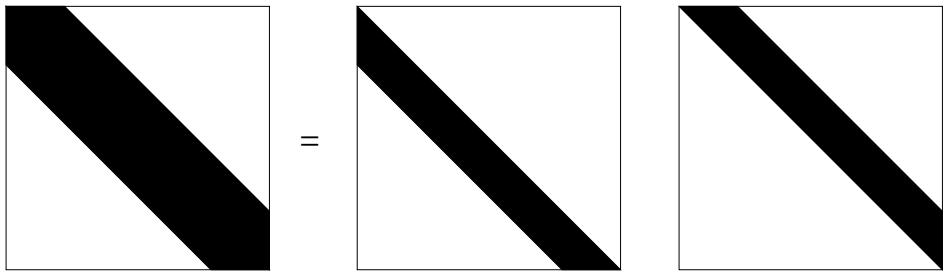
Bandwidth is preserved

Similarly, for an AR(p)-process

- $\boldsymbol{Q}$ have bandwidth p.
- $\boldsymbol{L}$ have lower-bandwidth p.

Theorem

Let $\boldsymbol{Q} > 0$ be a band matrix with bandwidth p and dimension n, then the Cholesky triangle of $\boldsymbol{Q}$ has (lower) bandwidth p.



...easy to modify existing Cholesky-factorisation code to use only entries where $|i - j| \leq p$.

Reorder the vertices

We can permute the vertexes;

> *select one of the n! possible permutations, define the corresponding permutation matrix* $\boldsymbol{P}$*, such that* $\boldsymbol{i}^P = \boldsymbol{P}\boldsymbol{i}$*, where* $\boldsymbol{i} = (1, \ldots, n)^T$*, is the new ordering of the vertexes.*

Chose $\boldsymbol{P}$, if possible, such that

$$\boldsymbol{Q}^P = \boldsymbol{P}\boldsymbol{Q}\boldsymbol{P}^T \tag{3}$$

is a band-matrix with a small bandwidth.

Reorder the vertices

We can permute the vertexes;

> *select one of the n! possible permutations, define the corresponding permutation matrix* $\boldsymbol{P}$*, such that* $\boldsymbol{i}^P = \boldsymbol{P}\boldsymbol{i}$*, where* $\boldsymbol{i} = (1, \ldots, n)^T$*, is the new ordering of the vertexes.*

Chose $\boldsymbol{P}$, if possible, such that

$$\boldsymbol{Q}^P = \boldsymbol{P}\boldsymbol{Q}\boldsymbol{P}^T \tag{3}$$

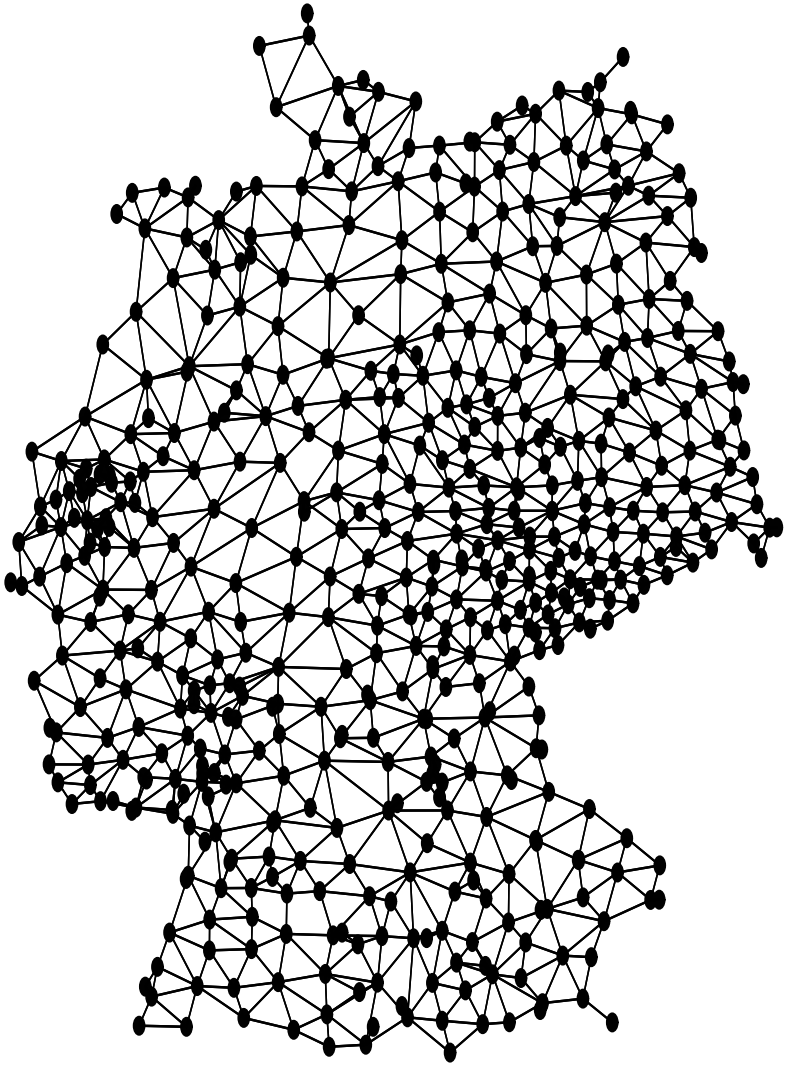
is a band-matrix with a small bandwidth.

- Impossible in general to obtain the optimal permutation, $n!$ is to large!
- A sub-optimal ordering will do as well.
- Solve $\boldsymbol{Q}\mu = \boldsymbol{b}$ as follows:
  - $\boldsymbol{b}^P = \boldsymbol{P}\boldsymbol{b}$.
  - Solve $\boldsymbol{Q}^P \mu^P = \boldsymbol{b}^P$
  - Map the solution back, $\mu = \boldsymbol{P}^T \mu^P$.

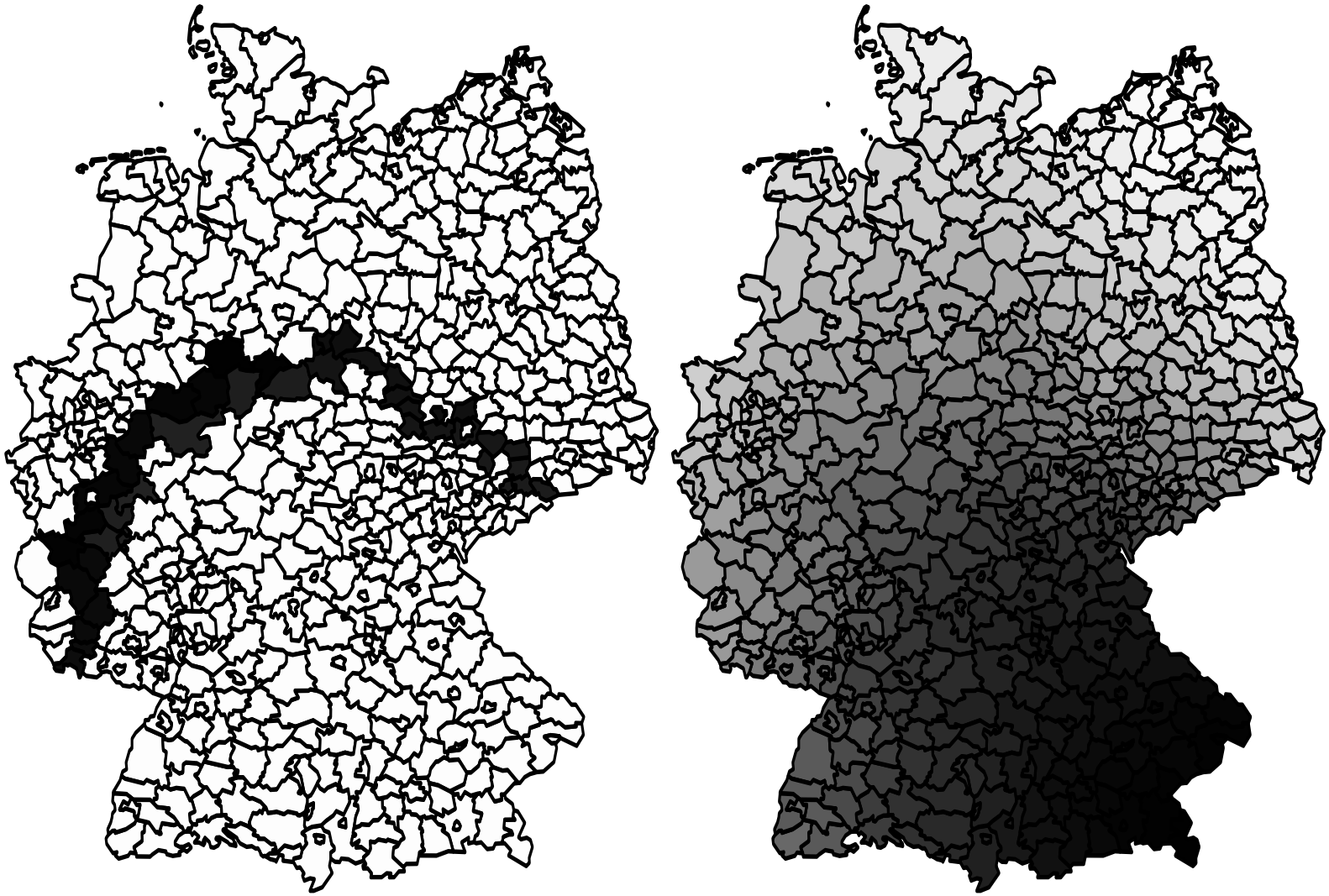
- Impossible in general to obtain the optimal permutation, $n!$ is to large!
- A sub-optimal ordering will do as well.
- Solve $\boldsymbol{Q}\mu = \boldsymbol{b}$ as follows:
  - $\boldsymbol{b}^P = \boldsymbol{P}\boldsymbol{b}$.
  - Solve $\boldsymbol{Q}^P \mu^P = \boldsymbol{b}^P$
  - Map the solution back, $\mu = \boldsymbol{P}^T \mu^P$.

- Impossible in general to obtain the optimal permutation, $n!$ is to large!
- A sub-optimal ordering will do as well.
- Solve $\boldsymbol{Q}\boldsymbol{\mu} = \boldsymbol{b}$ as follows:
  - $\boldsymbol{b}^P = \boldsymbol{P}\boldsymbol{b}$.
  - Solve $\boldsymbol{Q}^P \boldsymbol{\mu}^P = \boldsymbol{b}^P$
  - Map the solution back, $\boldsymbol{\mu} = \boldsymbol{P}^T \boldsymbol{\mu}^P$.

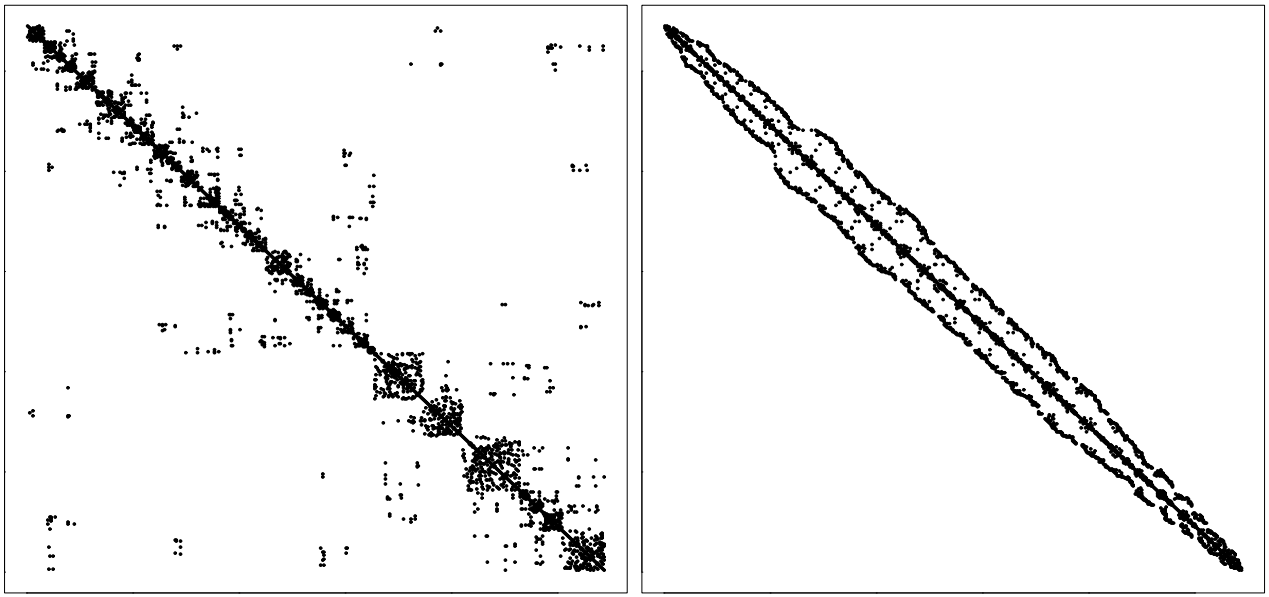
Reordering to band-matrices



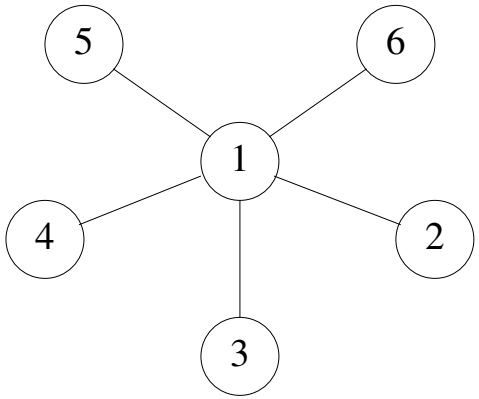
Reordering to band-matrices



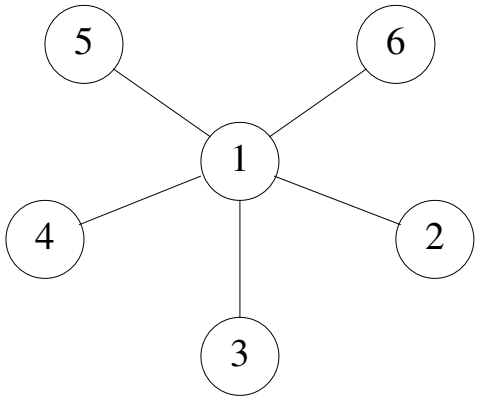
Reordering to band-matrices

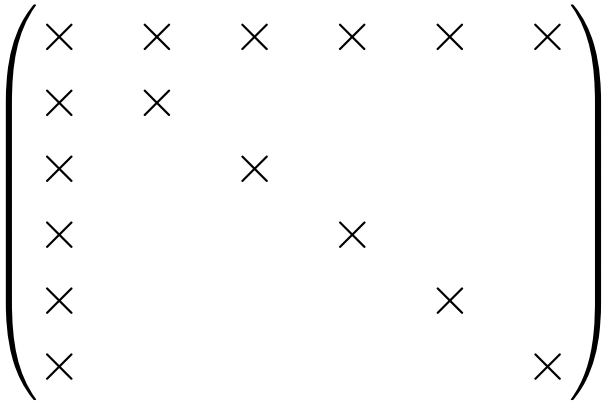


More optimal reordering schemes

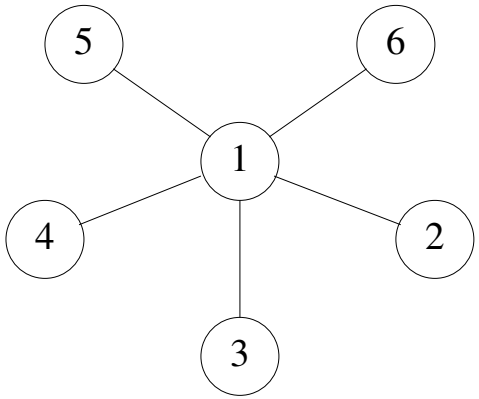


More optimal reordering schemes





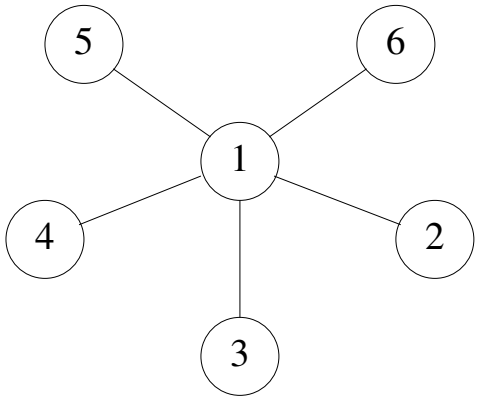
More optimal reordering schemes

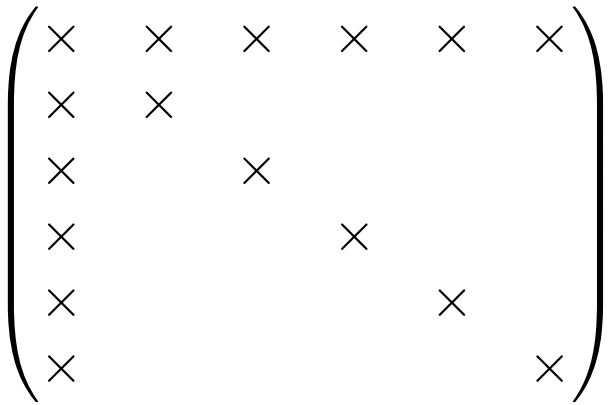


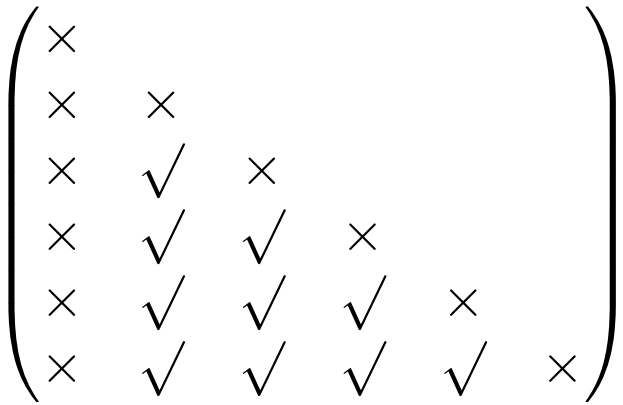
$$\begin{pmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & & & & \\ \times & & \times & & & \\ \times & & & \times & & \\ \times & & & & \times & \\ \times & & & & & \times \end{pmatrix} \quad \begin{pmatrix} \times & & & & & \\ \times & \times & & & & \\ \times & \surd & \times & & & \\ \times & \surd & \surd & \times & & \\ \times & \surd & \surd & \surd & \times & \\ \times & \surd & \surd & \surd & \surd & \times \end{pmatrix}$$

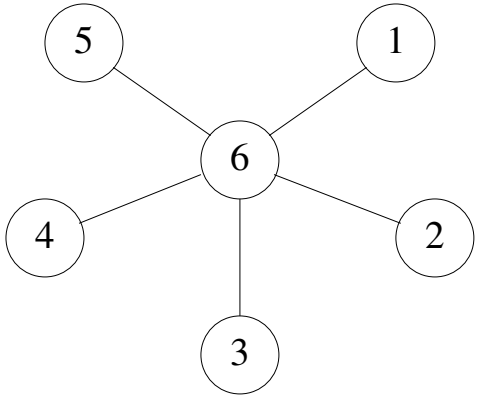
$$\begin{pmatrix} \times & & & & & \times \\ & \times & & & & \times \\ & & \times & & & \times \\ & & & \times & & \times \\ & & & & \times & \times \\ \times & \times & \times & \times & \times & \times \end{pmatrix} \quad \begin{pmatrix} \times & & & & & \\ & \times & & & & \\ & & \times & & & \\ & & & \times & & \\ & & & & \times & \\ \times & \times & \times & \times & \times & \times \end{pmatrix}$$

More optimal reordering schemes

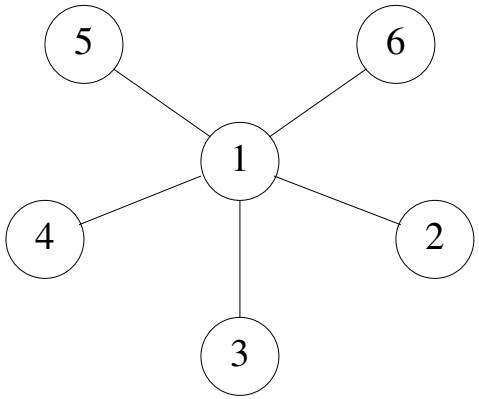




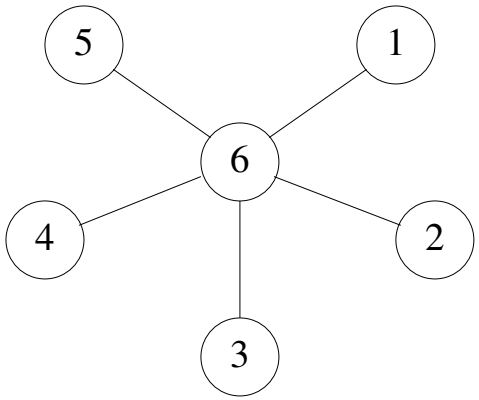




More optimal reordering schemes

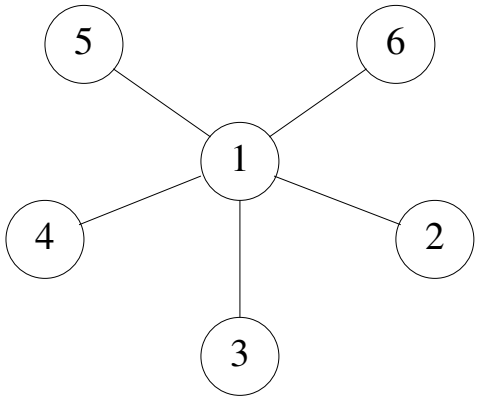


$$\begin{pmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & & & & \\ \times & & \times & & & \\ \times & & & \times & & \\ \times & & & & \times & \\ \times & & & & & \times \end{pmatrix} \quad \begin{pmatrix} \times & & & & & \\ \times & \times & & & & \\ \times & \surd & \times & & & \\ \times & \surd & \surd & \times & & \\ \times & \surd & \surd & \surd & \times & \\ \times & \surd & \surd & \surd & \surd & \times \end{pmatrix}$$

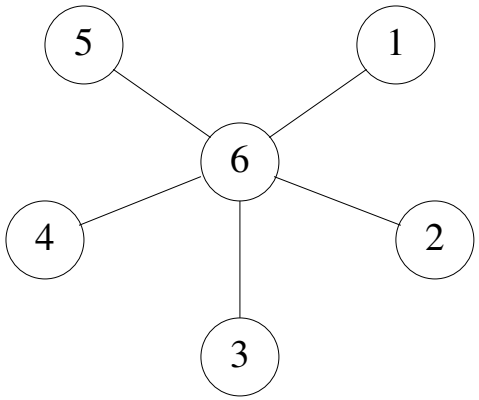


$$\begin{pmatrix} \times & & & & & \times \\ & \times & & & & \times \\ & & \times & & & \times \\ & & & \times & & \times \\ & & & & \times & \times \\ \times & \times & \times & \times & \times & \times \end{pmatrix} \quad \begin{pmatrix} \times & & & & & \\ & \times & & & & \\ & & \times & & & \\ & & & \times & & \\ & & & & \times & \\ & & & & & \times \end{pmatrix}$$

More optimal reordering schemes



$$\begin{pmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & & & & \\ \times & & \times & & & \\ \times & & & \times & & \\ \times & & & & \times & \\ \times & & & & & \times \end{pmatrix} \quad \begin{pmatrix} \times & & & & & \\ \times & \times & & & & \\ \times & \surd & \times & & & \\ \times & \surd & \surd & \times & & \\ \times & \surd & \surd & \surd & \times & \\ \times & \surd & \surd & \surd & \surd & \times \end{pmatrix}$$



$$\begin{pmatrix} \times & & & & & \times \\ & \times & & & & \times \\ & & \times & & & \times \\ & & & \times & & \times \\ & & & & \times & \times \\ \times & \times & \times & \times & \times & \times \end{pmatrix} \quad \begin{pmatrix} \times & & & & & \\ & \times & & & & \\ & & \times & & & \\ & & & \times & & \\ & & & & \times & \\ \times & \times & \times & \times & \times & \times \end{pmatrix}$$

Nested dissection reordering (I)

The idea generalise as follows.

- Select a (small) set of nodes whose removal divides the graph into two disconnected subgraphs of almost equal size.
- Order the nodes chosen *after* ordering all the nodes in both subgraphs.
- Apply this procedure recursively to the nodes in each subgraph.

Costs in the spatial case

- Factorisation $\mathcal{O}(n^{3/2})$
- Fill-in $\mathcal{O}(n \log n)$
- Optimal in the order sense.

Nested dissection reordering (I)

The idea generalise as follows.

- Select a (small) set of nodes whose removal divides the graph into two disconnected subgraphs of almost equal size.
- Order the nodes chosen *after* ordering all the nodes in both subgraphs.
- Apply this procedure recursively to the nodes in each subgraph.

Costs in the spatial case

- Factorisation $\mathcal{O}(n^{3/2})$
- Fill-in $\mathcal{O}(n \log n)$
- Optimal in the order sense.

Nested dissection reordering (I)

The idea generalise as follows.

- Select a (small) set of nodes whose removal divides the graph into two disconnected subgraphs of almost equal size.
- Order the nodes chosen *after* ordering all the nodes in both subgraphs.
- Apply this procedure recursively to the nodes in each subgraph.

Costs in the spatial case

- Factorisation $\mathcal{O}(n^{3/2})$
- Fill-in $\mathcal{O}(n \log n)$
- Optimal in the order sense.

Nested dissection reordering (I)

The idea generalise as follows.

- Select a (small) set of nodes whose removal divides the graph into two disconnected subgraphs of almost equal size.
- Order the nodes chosen *after* ordering all the nodes in both subgraphs.
- Apply this procedure recursively to the nodes in each subgraph.

Costs in the spatial case

- Factorisation $\mathcal{O}(n^{3/2})$
- Fill-in $\mathcal{O}(n \log n)$
- Optimal in the order sense.

Nested dissection reordering (I)

The idea generalise as follows.

- Select a (small) set of nodes whose removal divides the graph into two disconnected subgraphs of almost equal size.
- Order the nodes chosen *after* ordering all the nodes in both subgraphs.
- Apply this procedure recursively to the nodes in each subgraph.

Costs in the spatial case

- Factorisation $\mathcal{O}(n^{3/2})$
- Fill-in $\mathcal{O}(n \log n)$
- Optimal in the order sense.

Nested dissection reordering (I)

The idea generalise as follows.

- Select a (small) set of nodes whose removal divides the graph into two disconnected subgraphs of almost equal size.
- Order the nodes chosen *after* ordering all the nodes in both subgraphs.
- Apply this procedure recursively to the nodes in each subgraph.

Costs in the spatial case

- Factorisation $\mathcal{O}(n^{3/2})$
- Fill-in $\mathcal{O}(n \log n)$
- Optimal in the order sense.

Nested dissection reordering (I)

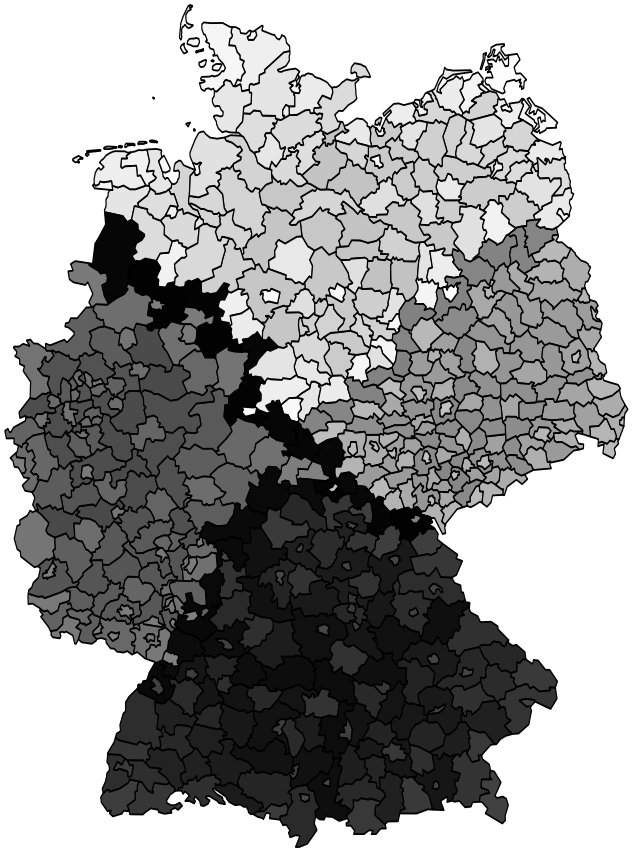
The idea generalise as follows.

- Select a (small) set of nodes whose removal divides the graph into two disconnected subgraphs of almost equal size.
- Order the nodes chosen *after* ordering all the nodes in both subgraphs.
- Apply this procedure recursively to the nodes in each subgraph.

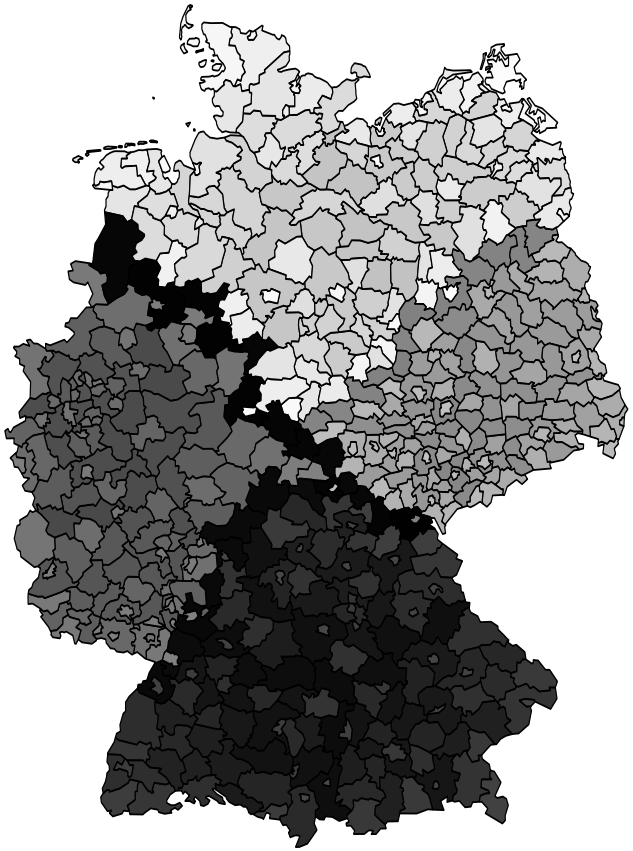
Costs in the spatial case

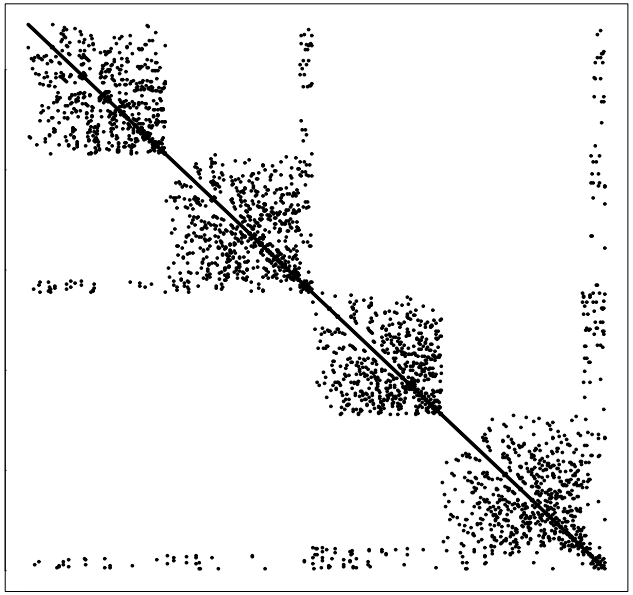
- Factorisation $\mathcal{O}(n^{3/2})$
- Fill-in $\mathcal{O}(n \log n)$
- Optimal in the order sense.

Nested dissection reordering (II)

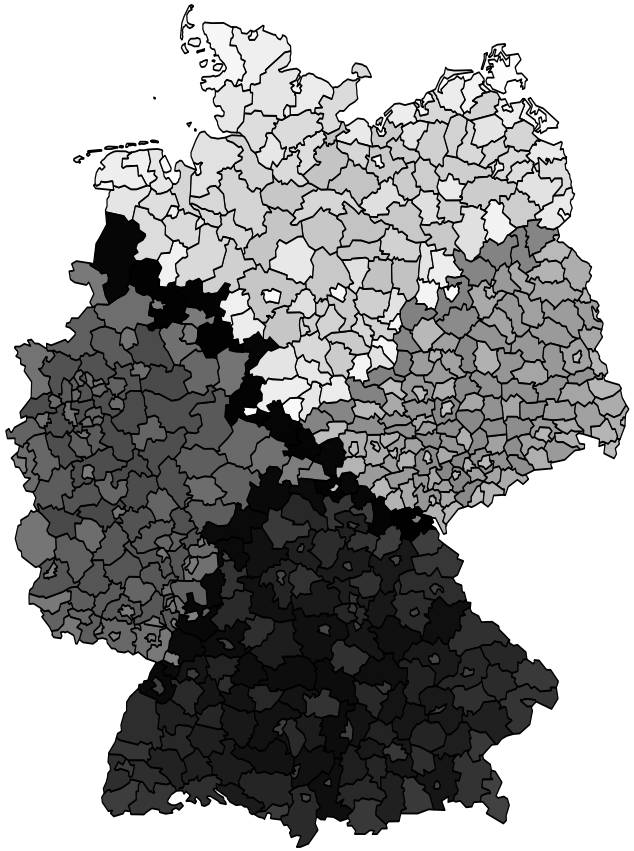


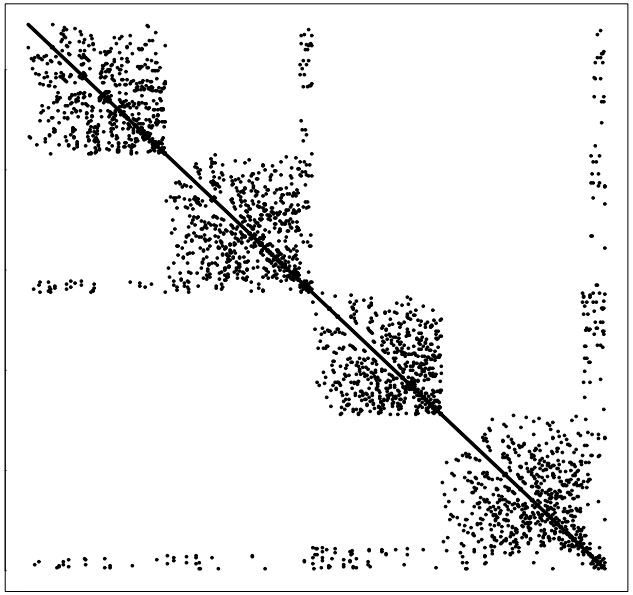
Nested dissection reordering (II)





Nested dissection reordering (II)





How to deal with sparse matrices?

- Use existing software.
- In `R` use (recomended) package `Matrix`, create sparse matrices with `sparseMatrix()` etc...

Part V

Marginal variance

Computing marginal variances for GMRFs*

Let

$$\boldsymbol{Q} = \boldsymbol{V}\boldsymbol{D}\boldsymbol{V}^T$$

- where $\boldsymbol{D}$ is a diagonal matrix, and
- $\boldsymbol{V}$ is a lower triangular matrix with ones on the diagonal.

The matrix identity

$$\boldsymbol{\Sigma} = \boldsymbol{D}^{-1}\boldsymbol{V}^{-1} + (\boldsymbol{I} - \boldsymbol{V}^T)\boldsymbol{\Sigma}$$

define *recursions* which can be used to compute

- $\text{Var}(x_i)$ and $\text{Cov}(x_i, x_j)$ for $i \sim j$

essentially without cost when the Cholesky triangle $\boldsymbol{L}$ is known (TFC '73).

Computing marginal variances for GMRFs*

Let

$$\boldsymbol{Q} = \boldsymbol{V}\boldsymbol{D}\boldsymbol{V}^T$$

- where $\boldsymbol{D}$ is a diagonal matrix, and
- $\boldsymbol{V}$ is a lower triangular matrix with ones on the diagonal.

The matrix identity

$$\boldsymbol{\Sigma} = \boldsymbol{D}^{-1}\boldsymbol{V}^{-1} + (\boldsymbol{I} - \boldsymbol{V}^T)\boldsymbol{\Sigma}$$

define *recursions* which can be used to compute

- $\text{Var}(x_i)$ and $\text{Cov}(x_i, x_j)$ for $i \sim j$

essentially without cost when the Cholesky triangle $\boldsymbol{L}$ is known (TFC '73).

FORMATION OF A SPARSE BUS IMPEDANCE MATRIX AND ITS APPLICATION TO SHORT CIRCUIT STUDY

by

Kazuhiro Takahashi, John Fagan and Mo-Shing Chen

ABSTRACT

This paper proposes a new computational method for the formation of a sparse bus impedance matrix. An algorithm is introduced which generates only the relevant terms of the bus impedance matrix for short circuit calculation. The algorithm is presented as it applies to the solution of short circuit problems. The method offers a decrease in execution time and a reduction in core memory requirements for the digital computer. These advantages are superior to conventional methods of short circuit calculation such as the ordinary Z matrix and the factorized Y matrix methods.

A small sample system is provided to illustrate the computational procedure. Application to practical large power systems has been proven by the development of a digital computer program based upon the proposed method.

The following considerations are made to extend the method for more practical usage; 1) Mutual coupling of transmission lines, 2) Short circuit currents more than 2 busses away, 3) Asymmetric fault analysis, 4) Modification of the matrix elements due to the network configuration change, and 5) Inversion of asymmetric sparse matrix.

$$I_{ss} = 1/Z_{ss} \quad (1)$$

$$I_{st} = I_{ss} \cdot (Z_{ss} - Z_{st})/\mathcal{Z}_{st} \quad (2)$$

where I_{ss} is the three phase short circuit current at the faulted bus s
I_{st} is the current of the connecting line (st) due to the fault
$\mathcal{Z}_{st}$ is the impedance of the connecting line (st)
Z_{ss} is the driving point impedance at the faulted bus s
Z_{st} is the transfer impedance of the faulted bus s to the adjacent bus t

These system parameters are normally complex. However, real numbers are used in this paper to illustrate the method.

The driving point and transfer impedances given in eqs. (1) and (2) are normally obtained by calculating all the terms of the bus impedance matrix. The bus impedance matrix, or simply Z matrix, is defined by the inverse of the bus admittance matrix Y . Both matrices are symmetric. The Y matrix is generally sparse for practical power systems while the Z matrix is full.

A small DC network is provided in Fig. 2, which

Statistical derivation

Recall for a zero mean GMRF that

$$x_i \mid x_{i+1}, \ldots, x_n \sim \mathcal{N}(-\frac{1}{L_{ii}} \sum_{k=i+1}^{n} L_{ki} x_k, 1/L_{ii}^2), \quad i = n, \ldots, 1.$$

provides a sequential representation of the GMRF backward in "time" i.

Multiply by $x_j, j \geq i$, and taking expectation yields

$$\Sigma_{ij} = \delta_{ij}/L_{ii}^2 - \frac{1}{L_{ii}} \sum_{k \in \mathcal{I}(i)}^{n} L_{ki}\Sigma_{kj}, \quad j \geq i,\ i = n, \ldots, 1,$$

where $\mathcal{I}(i)$ as those k where L_{ki} is non-zero,

$$\mathcal{I}(i) = \{k > i \ : \ L_{ki} \neq 0\}$$

and δ_{ij} is one if $i = j$ and zero otherwise.

We can use

$$\Sigma_{ij} = \delta_{ij}/L_{ii}^2 - \frac{1}{L_{ii}} \sum_{k \in \mathcal{I}(i)}^{n} L_{ki}\Sigma_{kj}, \quad j \geq i,\ i = n, \ldots, 1,$$

to compute Σ_{ij} for each ij:

- Outer loop $i = n, \ldots, 1$
- Inner loop $j = n, \ldots, i$

Example

Let $n = 3, \mathcal{I}(1) = \{2, 3\}, \mathcal{I}(2) = \{3\}$, then we get

$$\Sigma_{33} = \frac{1}{L_{33}^2} \qquad \Sigma_{23} = -\frac{1}{L_{22}}(L_{32}\Sigma_{33})$$

$$\Sigma_{22} = \frac{1}{L_{22}^2} - \frac{1}{L_{22}}(L_{32}\Sigma_{32}) \qquad \Sigma_{13} = -\frac{1}{L_{11}}(L_{21}\Sigma_{23} + L_{31}\Sigma_{33})$$

$$\Sigma_{12} = -\frac{1}{L_{11}}(L_{21}\Sigma_{22} + L_{31}\Sigma_{32}) \qquad \Sigma_{11} = \frac{1}{L_{11}^2} - \frac{1}{L_{11}}(L_{21}\Sigma_{21} + L_{31}\Sigma_{31})$$

where we also need to use that $\mathbf{\Sigma}$ is symmetric.

- Assume we want to compute all marginal variances.
- To do so, we need to compute Σ_{ij} (or Σ_{ji}) for all ij in some set $\mathcal{S}$.
- If the recursions can be solved by only computing Σ_{ij} for all $ij \in \mathcal{S}$ we say that the recursions are *solvable* using $\mathcal{S}$.

- Assume we want to compute all marginal variances.
- To do so, we need to compute Σ_{ij} (or Σ_{ji}) for all ij in some set $\mathcal{S}$.
- If the recursions can be solved by only computing Σ_{ij} for all $ij \in \mathcal{S}$ we say that the recursions are *solvable* using $\mathcal{S}$.

- Assume we want to compute all marginal variances.
- To do so, we need to compute Σ_{ij} (or Σ_{ji}) for all ij in some set $\mathcal{S}$.
- If the recursions can be solved by only computing Σ_{ij} for all $ij \in \mathcal{S}$ we say that the recursions are *solvable* using $\mathcal{S}$.

From

$$\Sigma_{ij} = \delta_{ij}/L_{ii}^2 - \frac{1}{L_{ii}} \sum_{k \in \mathcal{I}(i)}^{n} L_{ki}\Sigma_{kj}, \quad j \geq i,\ i = n, \ldots, 1, \tag{4}$$

it is evident that $\mathcal{S}$ must satisfy

$$ij \in \mathcal{S} \text{ and } k \in \mathcal{I}(i) \longrightarrow kj \in \mathcal{S} \tag{5}$$

we also need that $ii \in \mathcal{S}$ for $i = 1, \ldots, n$.

From

$$\Sigma_{ij} = \delta_{ij}/L_{ii}^2 - \frac{1}{L_{ii}} \sum_{k \in \mathcal{I}(i)}^{n} L_{ki}\Sigma_{kj}, \quad j \geq i,\ i = n, \ldots, 1, \tag{4}$$

it is evident that $\mathcal{S}$ must satisfy

$$ij \in \mathcal{S} \text{ and } k \in \mathcal{I}(i) \longrightarrow kj \in \mathcal{S} \tag{5}$$

we also need that $ii \in \mathcal{S}$ for $i = 1, \ldots, n$.

- $\mathcal{S} = \mathcal{V} \times \mathcal{V}$ is a valid set, but we want $|\mathcal{S}|$ to be small to avoid unnecessary computations.
- a minimal set depends however on the numerical values in $\boldsymbol{L}$, or $\boldsymbol{Q}$ implicitly.
- a slightly larger set, that contains the minimal, turn out to be the one used to compute $\boldsymbol{L}$!!!

Theorem (TFC'73)

$$\mathcal{S} = \{ij \in \mathcal{V} \times \mathcal{V} \ : \ j \geq i, \ i \text{ and } j \text{ are not separated by } F(i,j)\}$$

is solvable.

- $\mathcal{S} = \mathcal{V} \times \mathcal{V}$ is a valid set, but we want $|\mathcal{S}|$ to be small to avoid unnecessary computations.
- a minimal set depends however on the numerical values in $\boldsymbol{L}$, or $\boldsymbol{Q}$ implicitly.
- a slightly larger set, that contains the minimal, turn out to be the one used to compute $\boldsymbol{L}$!!!

Theorem (TFC'73)

$$\mathcal{S} = \{ij \in \mathcal{V} \times \mathcal{V} \ : \ j \geq i, \ i \text{ and } j \text{ are not separated by } F(i,j)\}$$

is solvable.

- $\mathcal{S} = \mathcal{V} \times \mathcal{V}$ is a valid set, but we want $|\mathcal{S}|$ to be small to avoid unnecessary computations.
- a minimal set depends however on the numerical values in $\boldsymbol{L}$, or $\boldsymbol{Q}$ implicitly.
- a slightly larger set, that contains the minimal, turn out to be the one used to compute $\boldsymbol{L}$!!!

Theorem (TFC'73)

$$\mathcal{S} = \{ij \in \mathcal{V} \times \mathcal{V} \ : \ j \geq i, \ i \text{ and } j \text{ are not separated by } F(i,j)\}$$

is solvable.

- $\mathcal{S} = \mathcal{V} \times \mathcal{V}$ is a valid set, but we want $|\mathcal{S}|$ to be small to avoid unnecessary computations.
- a minimal set depends however on the numerical values in $\boldsymbol{L}$, or $\boldsymbol{Q}$ implicitly.
- a slightly larger set, that contains the minimal, turn out to be the one used to compute $\boldsymbol{L}$!!!

Theorem (TFC'73)

$$\mathcal{S} = \{ij \in \mathcal{V} \times \mathcal{V} \ : \ j \geq i, \ i \text{ and } j \text{ are not separated by } F(i,j)\}$$

is solvable.

Main idea of the proof:

$$ij \in \mathcal{S} \text{ and } k \in \mathcal{I}(i) \longrightarrow kj \in \mathcal{S}, \qquad i < j, \quad k < j \tag{6}$$

- $ij \in \mathcal{S}$, says there is a path from i to j where all nodes $\leq i$.
- $k \in \mathcal{I}(i)$, says there is path from i to k where all nodes $\leq i$.
- then there must be some path from k to i to j where all nodes are $\leq i$, hence $kj \in \mathcal{S}$.

Main idea of the proof:

$$ij \in \mathcal{S} \text{ and } k \in \mathcal{I}(i) \longrightarrow kj \in \mathcal{S}, \qquad i < j, \quad k < j \tag{6}$$

- $ij \in \mathcal{S}$, says there is a path from i to j where all nodes $\leq i$.
- $k \in \mathcal{I}(i)$, says there is path from i to k where all nodes $\leq i$.
- then there must be some path from k to i to j where all nodes are $\leq i$, hence $kj \in \mathcal{S}$.

Main idea of the proof:

$$ij \in \mathcal{S} \text{ and } k \in \mathcal{I}(i) \longrightarrow kj \in \mathcal{S}, \qquad i < j, \quad k < j \tag{6}$$

- $ij \in \mathcal{S}$, says there is a path from i to j where all nodes $\leq i$.
- $k \in \mathcal{I}(i)$, says there is path from i to k where all nodes $\leq i$.
- then there must be some path from k to i to j where all nodes are $\leq i$, hence $kj \in \mathcal{S}$.

Interpretation of $\mathcal{S}$

- $\mathcal{S}$ is the set of all possible non-zero elements in $\boldsymbol{L}$ based on $\mathcal{G}$ only.
- this is the set of L_{ji}'s that are computed when computing $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$.
- since $L_{ji} \neq 0$ in general when $i \sim j$, then we compute also $\text{Cov}(x_i, x_j)$ for $i \sim j$.
- some of the L_{ij}'s might turn out to be zero depending on the conditional independence properties of the marginal density for $\boldsymbol{x}_{i:n}$ for $i = n, \ldots, 1$.

Interpretation of $\mathcal{S}$

- $\mathcal{S}$ is the set of all possible non-zero elements in $\boldsymbol{L}$ based on $\mathcal{G}$ only.
- this is the set of L_{ji}'s that are computed when computing $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$.
- since $L_{ji} \neq 0$ in general when $i \sim j$, then we compute also $\text{Cov}(x_i, x_j)$ for $i \sim j$.
- some of the L_{ij}'s might turn out to be zero depending on the conditional independence properties of the marginal density for $\boldsymbol{x}_{i:n}$ for $i = n, \ldots, 1$.

Interpretation of $\mathcal{S}$

- $\mathcal{S}$ is the set of all possible non-zero elements in $\boldsymbol{L}$ based on $\mathcal{G}$ only.
- this is the set of L_{ji}'s that are computed when computing $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$.
- since $L_{ji} \neq 0$ in general when $i \sim j$, then we compute also $\text{Cov}(x_i, x_j)$ for $i \sim j$.
- some of the L_{ij}'s might turn out to be zero depending on the conditional independence properties of the marginal density for $\boldsymbol{x}_{i:n}$ for $i = n, \ldots, 1$.

Interpretation of $\mathcal{S}$

- $\mathcal{S}$ is the set of all possible non-zero elements in $\boldsymbol{L}$ based on $\mathcal{G}$ only.
- this is the set of L_{ji}'s that are computed when computing $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$.
- since $L_{ji} \neq 0$ in general when $i \sim j$, then we compute also $\text{Cov}(x_i, x_j)$ for $i \sim j$.
- some of the L_{ij}'s might turn out to be zero depending on the conditional independence properties of the marginal density for $\boldsymbol{x}_{i:n}$ for $i = n, \ldots, 1$.

General algorithm

for $i = n, \ldots, 1$

  for decreasing j *in* $\mathcal{I}(i)$

    compute Σ_{ij} *from eq. (4)*

band matrices

$$
\begin{aligned}
&\textit{for } i = n, \ldots, 1 \\
&\qquad \textit{for } j = \min(i + b_w, n), \ldots, i \\
&\qquad\qquad \textit{compute } \Sigma_{ij} \textit{ from eq. (4).}
\end{aligned}
$$

equivalent to Kalman-recursions for smoothing.

Part VI

INLA

Latent Gaussian models

Latent Gaussian models is a model of the following form

- Observed data $\boldsymbol{y}$, $y_i|x_i \sim \pi(y_i|x_i, \boldsymbol{\theta})$
- Latent Gaussian field $\boldsymbol{x} \sim \mathcal{N}(\cdot, \boldsymbol{Q}(\boldsymbol{\theta})^{-1})$
- Hyperparameters $\boldsymbol{\theta}$
  - variability
  - length/strength of dependence
  - parameters in the likelihood

$$\pi(\boldsymbol{x}, \boldsymbol{\theta} \mid \boldsymbol{y}) \propto \pi(\boldsymbol{\theta})\, \pi(\boldsymbol{x} \mid \boldsymbol{\theta}) \prod_{i \in \mathcal{I}} \pi(y_i \mid x_i, \boldsymbol{\theta})$$

Latent Gaussian models

Latent Gaussian models is a model of the following form

- Observed data $\boldsymbol{y}$, $y_i|x_i \sim \pi(y_i|x_i, \boldsymbol{\theta})$
- Latent Gaussian field $\boldsymbol{x} \sim \mathcal{N}(\cdot, \boldsymbol{Q}(\boldsymbol{\theta})^{-1})$
- Hyperparameters $\boldsymbol{\theta}$
  - variability
  - length/strength of dependence
  - parameters in the likelihood

$$\pi(\boldsymbol{x}, \boldsymbol{\theta} \mid \boldsymbol{y}) \propto \pi(\boldsymbol{\theta})\, \pi(\boldsymbol{x} \mid \boldsymbol{\theta}) \prod_{i \in \mathcal{I}} \pi(y_i \mid x_i, \boldsymbol{\theta})$$

Latent Gaussian models

Latent Gaussian models is a model of the following form

- Observed data $\boldsymbol{y}$, $y_i|x_i \sim \pi(y_i|x_i, \boldsymbol{\theta})$
- Latent Gaussian field $\boldsymbol{x} \sim \mathcal{N}(\cdot, \boldsymbol{Q}(\boldsymbol{\theta})^{-1})$
- Hyperparameters $\boldsymbol{\theta}$
  - variability
  - length/strength of dependence
  - parameters in the likelihood

$$\pi(\boldsymbol{x}, \boldsymbol{\theta} \mid \boldsymbol{y}) \propto \pi(\boldsymbol{\theta})\, \pi(\boldsymbol{x} \mid \boldsymbol{\theta}) \prod_{i \in \mathcal{I}} \pi(y_i \mid x_i, \boldsymbol{\theta})$$

Latent Gaussian models

Latent Gaussian models is a model of the following form

- Observed data $\boldsymbol{y}$, $y_i|x_i \sim \pi(y_i|x_i, \boldsymbol{\theta})$
- Latent Gaussian field $\boldsymbol{x} \sim \mathcal{N}(\cdot, \boldsymbol{Q}(\boldsymbol{\theta})^{-1})$
- Hyperparameters $\boldsymbol{\theta}$
  - variability
  - length/strength of dependence
  - parameters in the likelihood

$$\pi(\boldsymbol{x}, \boldsymbol{\theta} \mid \boldsymbol{y}) \propto \pi(\boldsymbol{\theta})\, \pi(\boldsymbol{x} \mid \boldsymbol{\theta}) \prod_{i \in \mathcal{I}} \pi(y_i \mid x_i, \boldsymbol{\theta})$$

Tasks

Compute from

$$\pi(\boldsymbol{x}, \boldsymbol{\theta} \mid \boldsymbol{y}) \propto \pi(\boldsymbol{\theta})\, \pi(\boldsymbol{x} \mid \boldsymbol{\theta}) \prod_{i \in \mathcal{I}} \pi(y_i \mid x_i)$$

the posterior marginals:

$$\pi(x_i \mid \boldsymbol{y}), \qquad \text{for some or all } i$$

and/or

$$\pi(\theta_i \mid \boldsymbol{y}), \qquad \text{for some or all } i$$

Our approach: Approximate Bayesian Inference

- Can we compute (approximate) marginals directly without using MCMC?
- YES!
- Gain
  - **Huge speedup & accuracy**
  - **The ability to treat latent Gaussian models properly ;-)**

Our approach: Approximate Bayesian Inference

- Can we compute (approximate) marginals directly without using MCMC?
- YES!
- Gain
  - **Huge speedup & accuracy**
  - **The ability to treat latent Gaussian models properly ;-)**

Our approach: Approximate Bayesian Inference

- Can we compute (approximate) marginals directly without using MCMC?
- YES!
- Gain
  - **Huge speedup & accuracy**
  - **The ability to treat latent Gaussian models properly ;-)**

Smoothing noisy observations (I)

Observations

$$y_i = m(i) + \epsilon_i, \qquad i = 1, \ldots, n$$

for Gaussian iid noise ϵ_i with *known* precision.

Will assume $m(i)$ is a smooth function wrt i

Smoothing noisy observations (I)

Observations

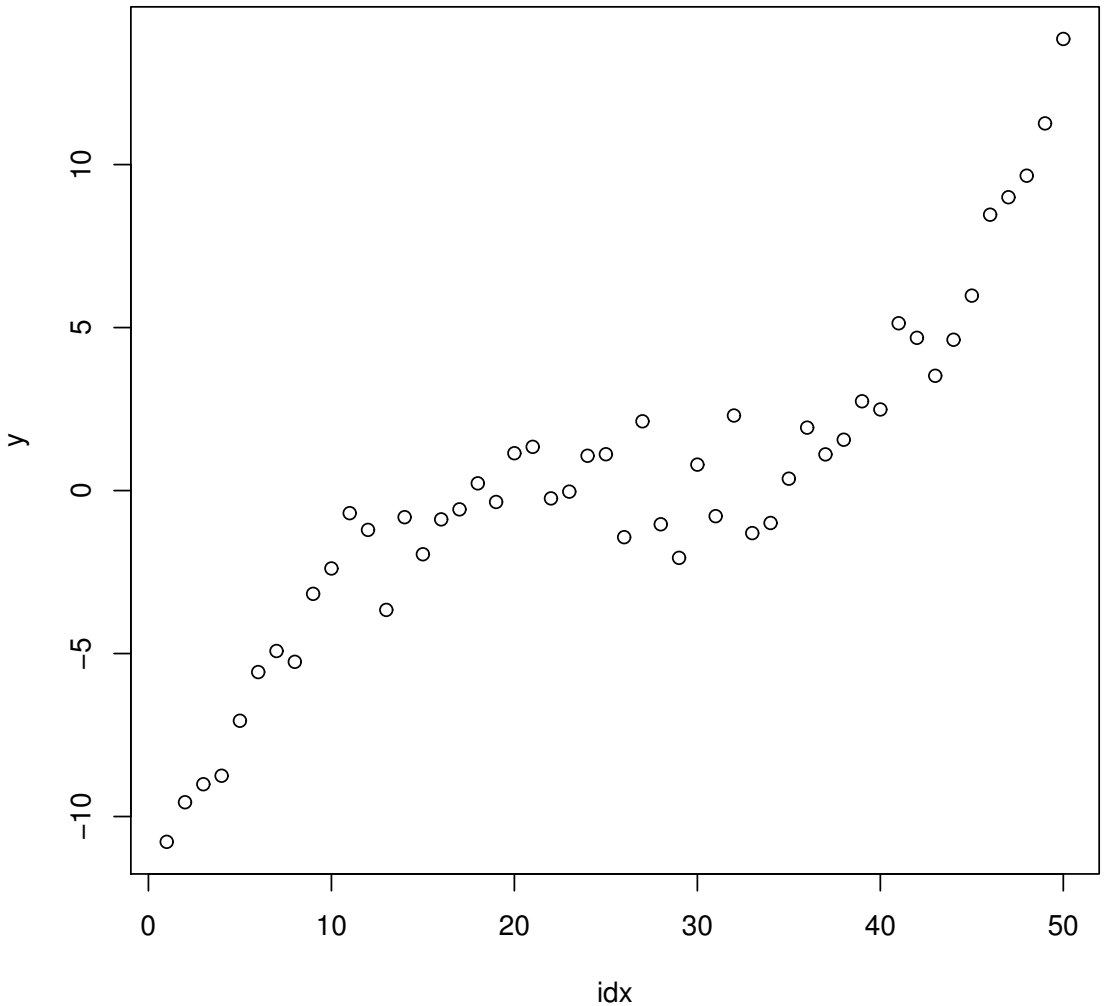
$$y_i = m(i) + \epsilon_i, \qquad i = 1, \ldots, n$$

for Gaussian iid noise ϵ_i with *known* precision.

Will assume $m(i)$ is a smooth function wrt i

Smoothing noisy observations (II)


```
n = 50
idx = 1:n
fun = 100*((idx-n/2)/n)^3
y = fun + rnorm(n)
plot(idx, y)
```



Smoothing noisy observations (III)

Likelihood Gaussian observations with known precision

$$y_i | x_i, \theta \sim \mathcal{N}(x_i, \tau_0)$$

Latent A Gaussian model for the smooth function[2]

$$\pi(\boldsymbol{x}|\theta) \propto \theta^{(n-2)/2} \exp\left(-\frac{\theta}{2}\sum_{i=2}^{n}(x_i - 2x_{i-1} + x_{i-2})^2\right)$$

Hyperparameter The smoothing parameter θ which we assign a $\Gamma(a, b)$ prior

$$\pi(\theta) \propto \theta^{a-1} \exp(-b\theta), \quad \theta > 0$$

[2] `model="rw2"`

Smoothing noisy observations (III)

Likelihood Gaussian observations with known precision

$$y_i | x_i, \theta \sim \mathcal{N}(x_i, \tau_0)$$

Latent A Gaussian model for the smooth function[2]

$$\pi(\boldsymbol{x}|\theta) \propto \theta^{(n-2)/2} \exp\left(-\frac{\theta}{2}\sum_{i=2}^{n}(x_i - 2x_{i-1} + x_{i-2})^2\right)$$

Hyperparameter The smoothing parameter θ which we assign a $\Gamma(a, b)$ prior

$$\pi(\theta) \propto \theta^{a-1} \exp(-b\theta), \quad \theta > 0$$

[2] `model="rw2"`

Smoothing noisy observations (III)

Likelihood Gaussian observations with known precision

$$y_i|x_i, \theta \sim \mathcal{N}(x_i, \tau_0)$$

Latent A Gaussian model for the smooth function[2]

$$\pi(\boldsymbol{x}|\theta) \propto \theta^{(n-2)/2} \exp\left(-\frac{\theta}{2}\sum_{i=2}^{n}(x_i - 2x_{i-1} + x_{i-2})^2\right)$$

Hyperparameter The smoothing parameter θ which we assign a $\Gamma(a, b)$ prior

$$\pi(\theta) \propto \theta^{a-1} \exp(-b\theta), \quad \theta > 0$$

[2] `model="rw2"`

Smoothing noisy observations (IV)

Since

$$\boldsymbol{x}, \boldsymbol{y}|\theta \sim \mathcal{N}(\cdot, \cdot)$$

we can compute (numerically) all marginals, using that

$$\pi(\theta|\boldsymbol{y}) \propto \frac{\overbrace{\pi(\boldsymbol{x}, \boldsymbol{y}|\theta)}^{\text{Gaussian}} \pi(\theta)}{\underbrace{\pi(\boldsymbol{x}|\boldsymbol{y}, \theta)}_{\text{Gaussian}}}$$

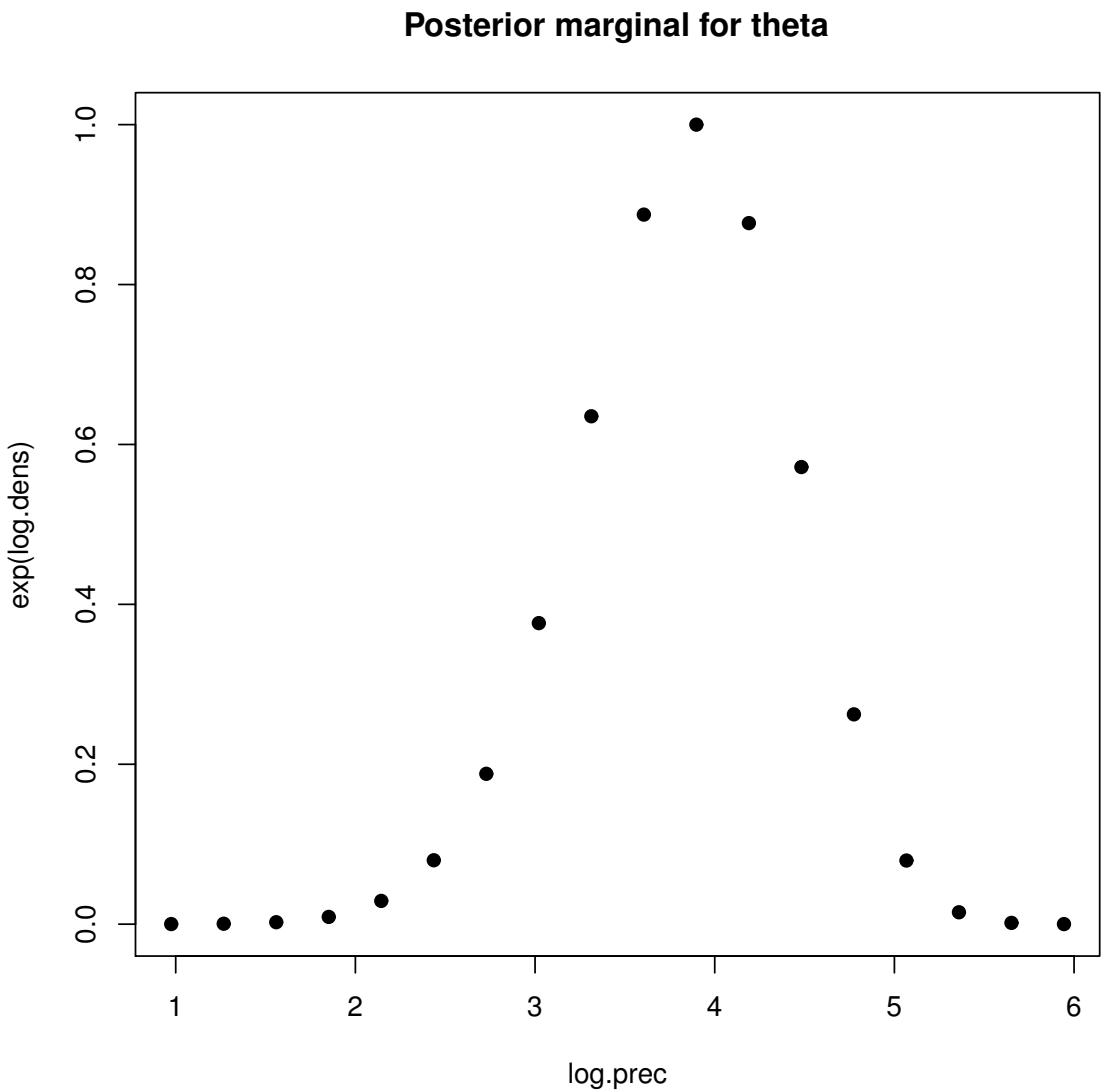
Smoothing noisy observations (IV)

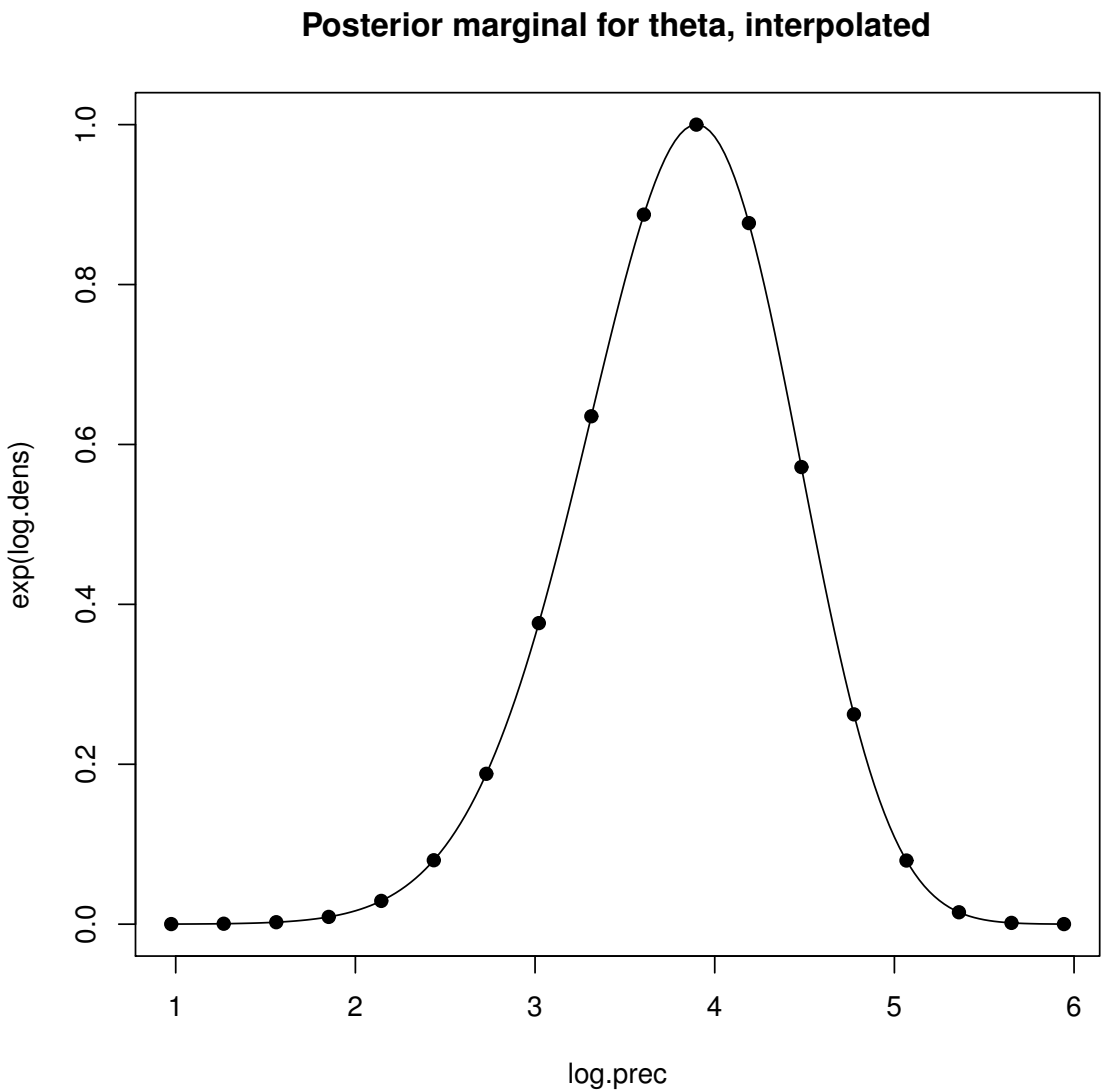
Since

$$\boldsymbol{x}, \boldsymbol{y}|\theta \sim \mathcal{N}(\cdot, \cdot)$$

we can compute (numerically) all marginals, using that

$$\pi(\theta|\boldsymbol{y}) \propto \frac{\overbrace{\pi(\boldsymbol{x}, \boldsymbol{y}|\theta)}^{\text{Gaussian}} \pi(\theta)}{\underbrace{\pi(\boldsymbol{x}|\boldsymbol{y}, \theta)}_{\text{Gaussian}}}$$


Posterior marginal for theta
exp(log.dens)
0.0
0.2
0.4
0.6
0.8
1.0
1
2
3
4
5
6
log.prec


Posterior marginal for theta, interpolated
exp(log.dens)
0.0
0.2
0.4
0.6
0.8
1.0
1
2
3
4
5
6
log.prec

Smoothing noisy observations (V)

$$\boldsymbol{x}|\boldsymbol{y},\theta \sim \mathcal{N}(\cdot,\cdot)$$

so that

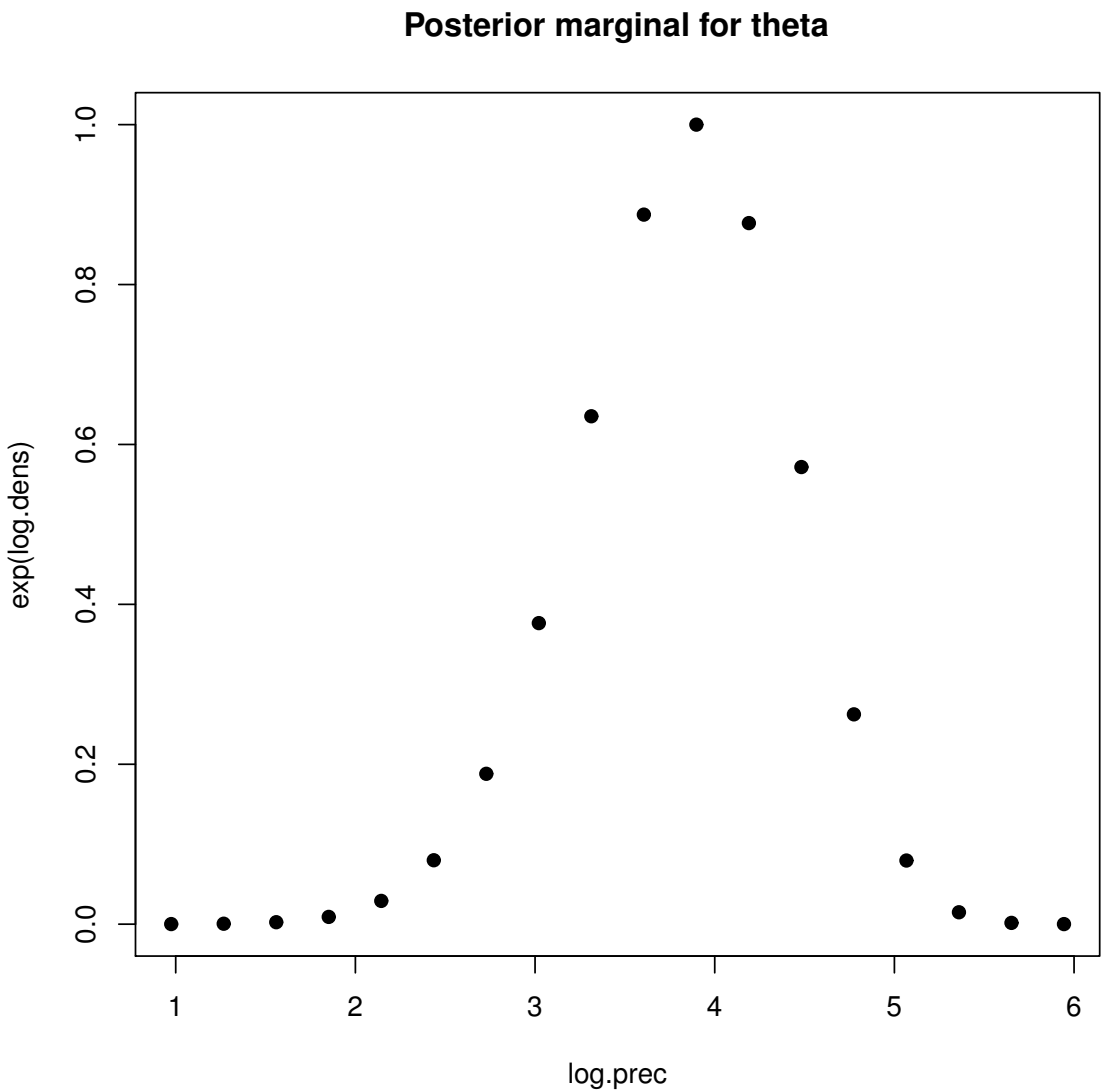
$$\pi(x_i|\boldsymbol{y}) = \int \underbrace{\pi(x_i|\theta,\boldsymbol{y})}_{\text{Gaussian}} \pi(\theta|\boldsymbol{y})\, d\theta$$

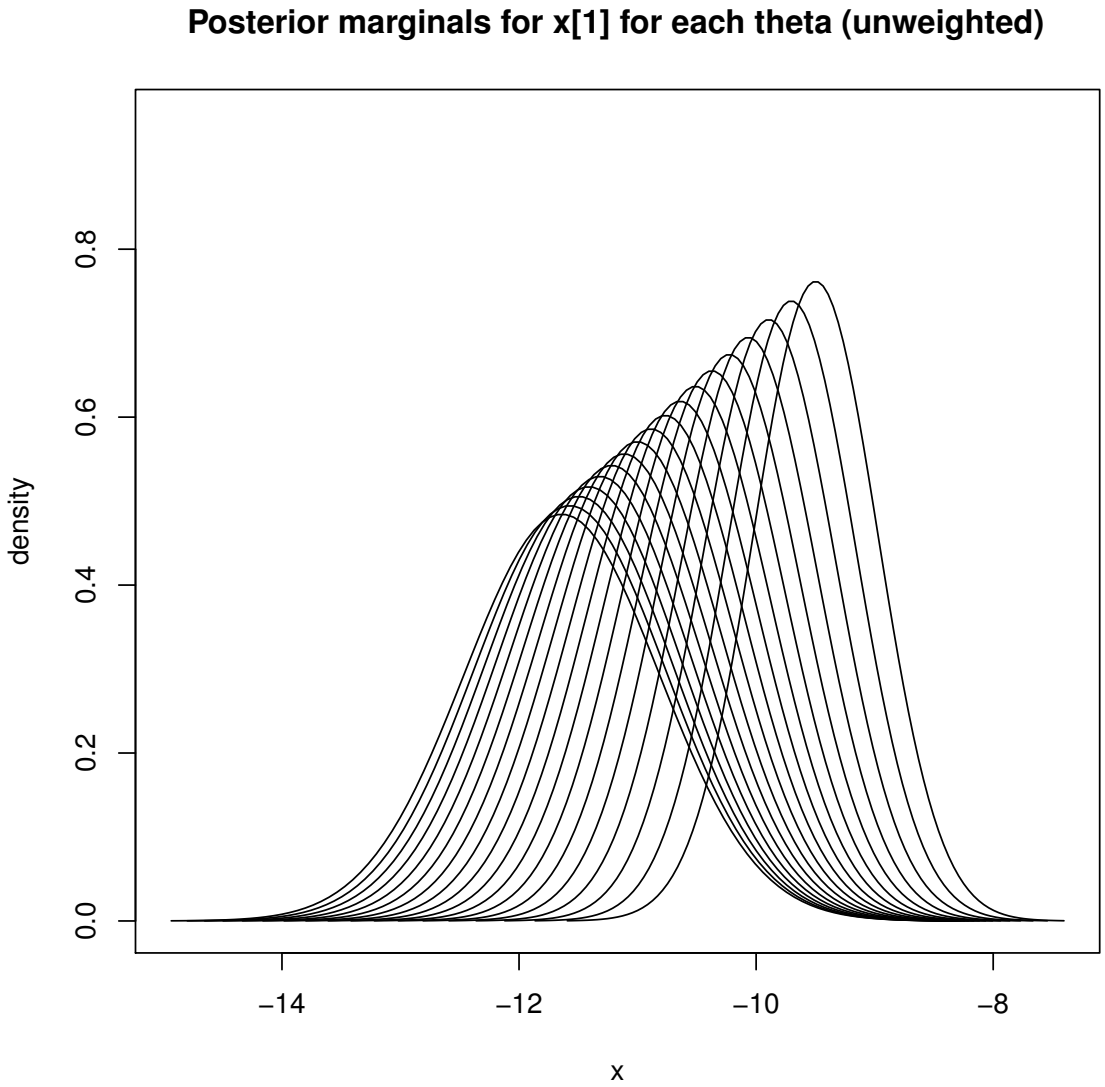
Smoothing noisy observations (V)

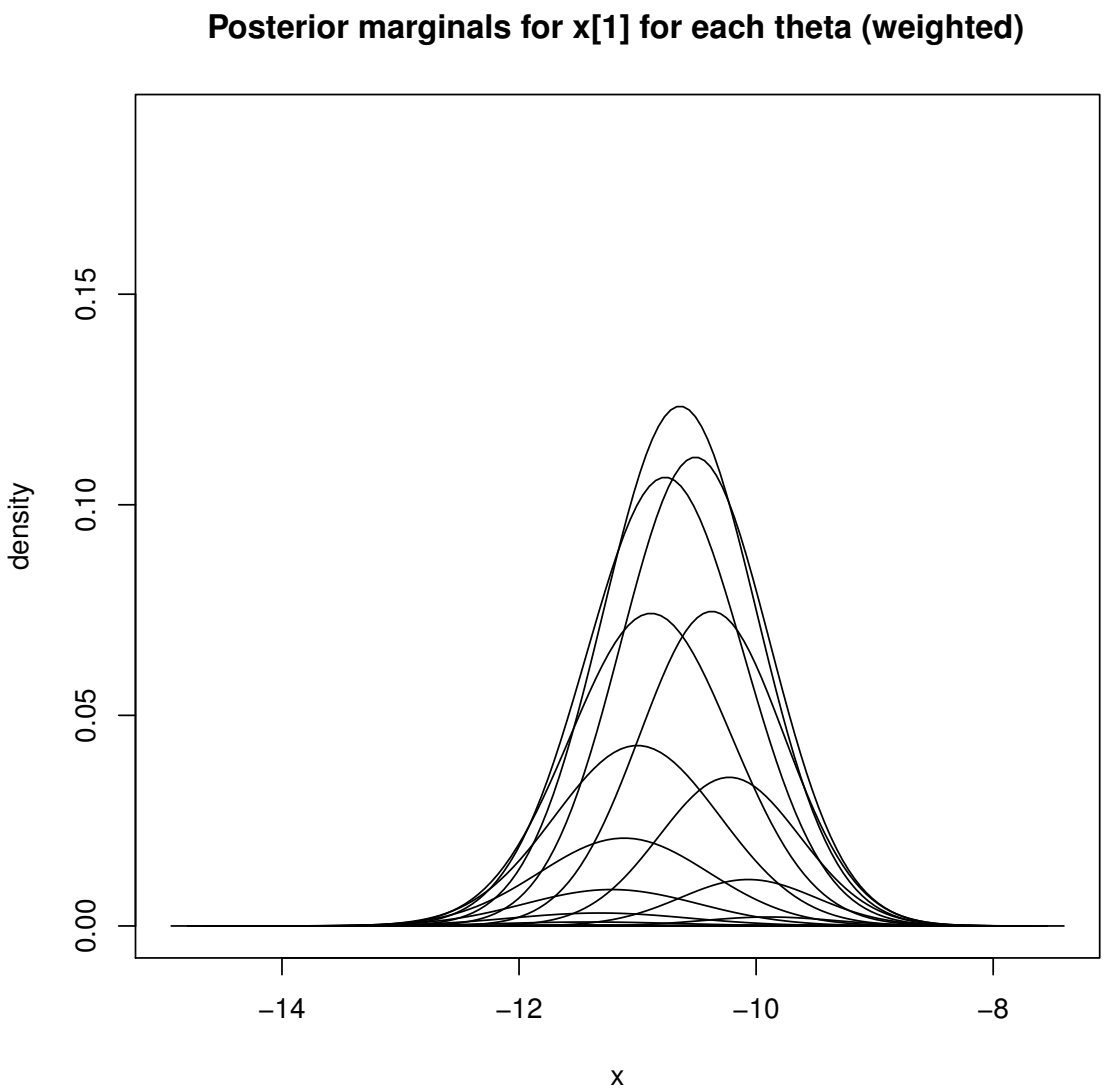
$$\boldsymbol{x}|\boldsymbol{y},\theta \;\sim\; \mathcal{N}(\cdot,\cdot)$$

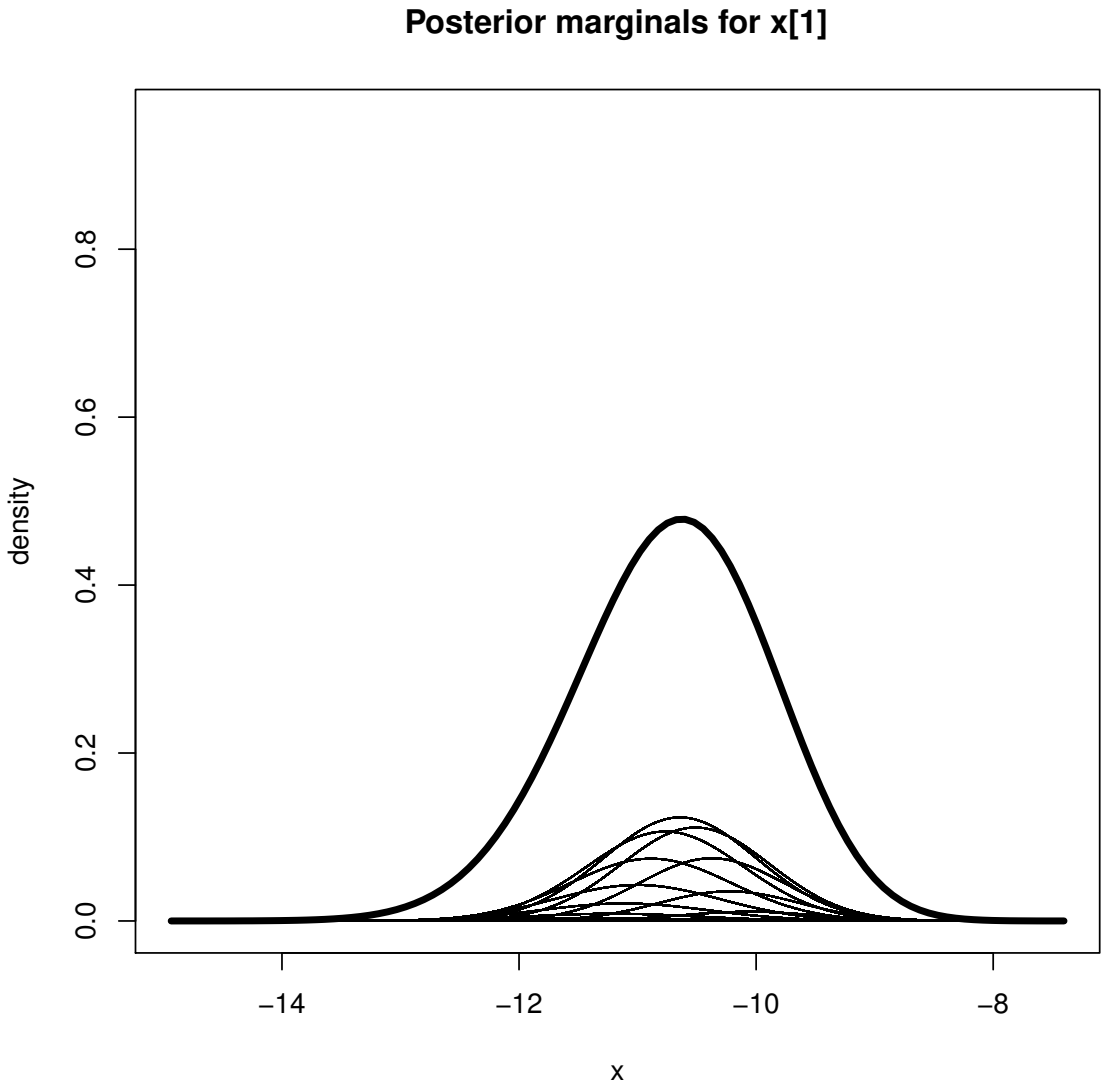
so that

$$\pi(x_i|\boldsymbol{y}) = \int \underbrace{\pi(x_i|\theta,\boldsymbol{y})}_{\text{Gaussian}}\; \pi(\theta|\boldsymbol{y})\; d\theta$$


Posterior marginal for theta
exp(log.dens)
0.0
0.2
0.4
0.6
0.8
1.0
1
2
3
4
5
6
log.prec


Posterior marginals for x[1] for each theta (unweighted)
density
0.0
0.2
0.4
0.6
0.8
−14
−12
−10
−8
x


Posterior marginals for x[1] for each theta (weighted)
density
0.00
0.05
0.10
0.15
−14
−12
−10
−8
x


Posterior marginals for x[1]
density
0.0
0.2
0.4
0.6
0.8
−14
−12
−10
−8
x

Extensions

This is the basic idea behind INLA. It is really really simple.

However, we need to extend this basic idea so we can deal with

- More than one hyperparameter
- Non-Gaussian observations

Complications... Mostly practical

Extensions

This is the basic idea behind INLA. It is really really simple.

However, we need to extend this basic idea so we can deal with

- More than one hyperparameter
- Non-Gaussian observations

Complications... Mostly practical

Extensions

This is the basic idea behind INLA. It is really really simple.

However, we need to extend this basic idea so we can deal with

- More than one hyperparameter
- Non-Gaussian observations

Complications... Mostly practical

Extensions

This is the basic idea behind INLA. It is really really simple.

However, we need to extend this basic idea so we can deal with

- More than one hyperparameter
- Non-Gaussian observations

Complications... Mostly practical

Extensions

This is the basic idea behind INLA. It is really really simple.

However, we need to extend this basic idea so we can deal with

- More than one hyperparameter
- Non-Gaussian observations

Complications... Mostly practical

More than one hyperparameter

Step I Explore $\pi(\boldsymbol{\theta}|\boldsymbol{y})$

- Locate the mode
- Use the Hessian to construct new variables
- Grid-search

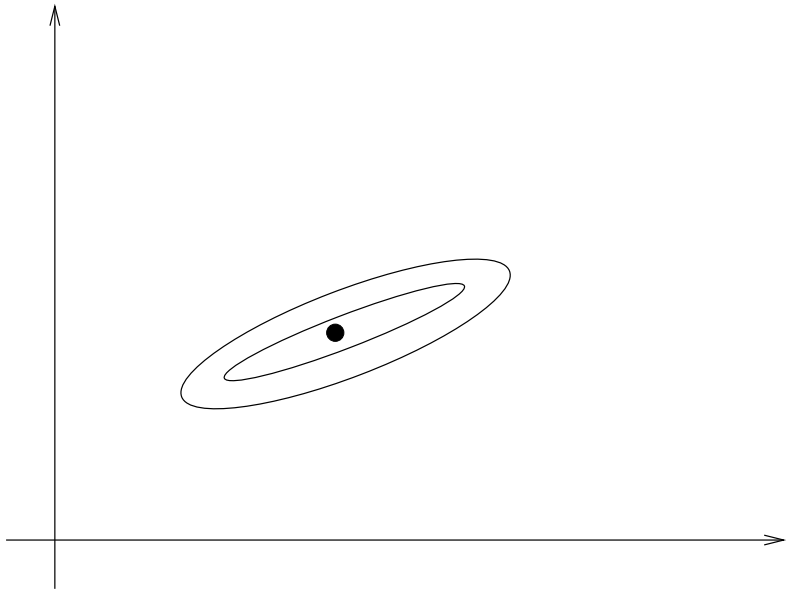
Step II From these integration points++, approximate the marginals for each θ_i

More than one hyperparameter

Step I Explore $\pi(\boldsymbol{\theta}|\boldsymbol{y})$

- Locate the mode
- Use the Hessian to construct new variables
- Grid-search

Step II From these integration points++, approximate the marginals for each θ_i

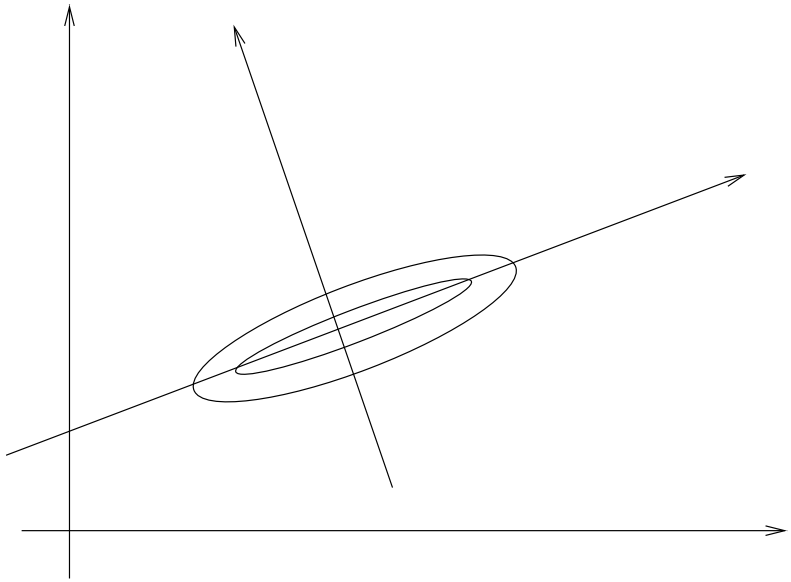


More than one hyperparameter

Step I Explore $\pi(\boldsymbol{\theta}|\boldsymbol{y})$

- Locate the mode
- Use the Hessian to construct new variables
- Grid-search

Step II From these integration points++, approximate the marginals for each θ_i

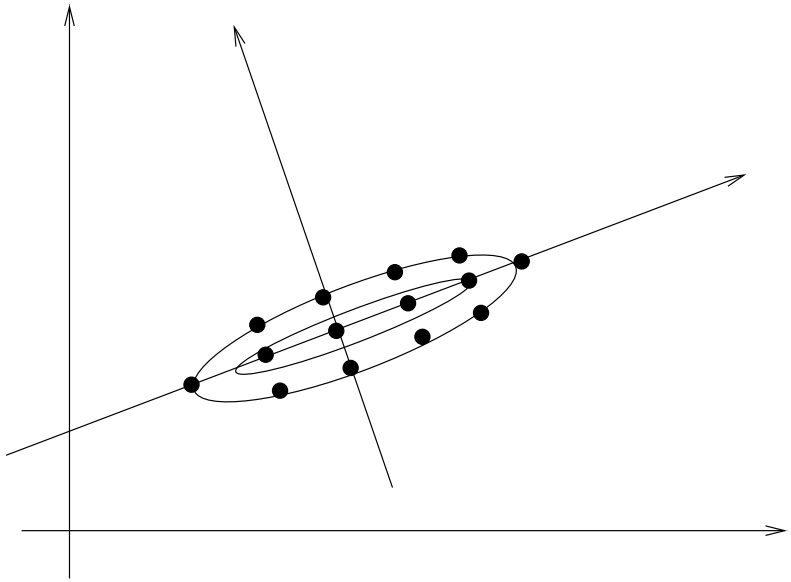


More than one hyperparameter

Step I Explore $\pi(\boldsymbol{\theta}|\boldsymbol{y})$

- Locate the mode
- Use the Hessian to construct new variables
- Grid-search

Step II From these integration points++, approximate the marginals for each θ_i

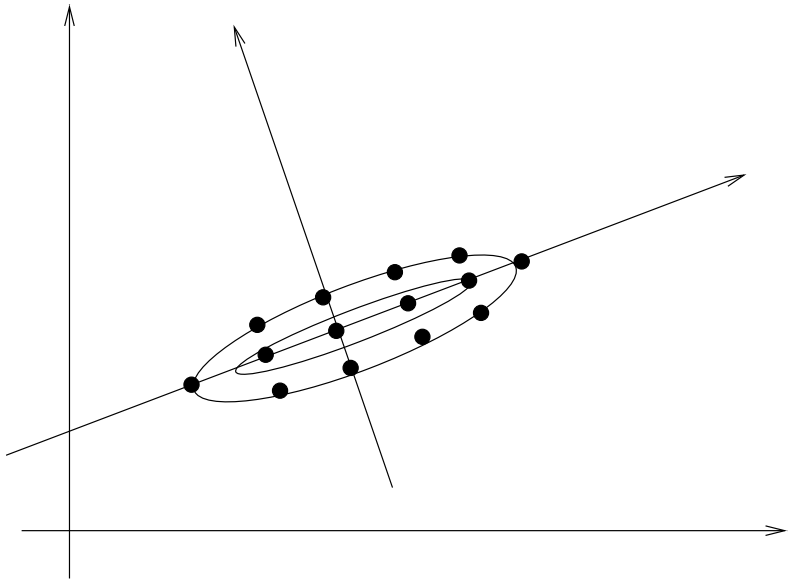


More than one hyperparameter

Step I Explore $\pi(\boldsymbol{\theta}|\boldsymbol{y})$

- Locate the mode
- Use the Hessian to construct new variables
- Grid-search

Step II From these integration points++, approximate the marginals for each θ_i



Non-Gaussian observations (I)

In this case

$$\pi(\theta|\boldsymbol{y}) \propto \frac{\overbrace{\pi(\boldsymbol{x},\boldsymbol{y}|\theta)}^{\text{Non-Gaussian, BUT KNOWN}}\ \pi(\theta)}{\underbrace{\pi(\boldsymbol{x}|\boldsymbol{y},\theta)}_{\text{Non-Gaussian and UNKNOWN}}}$$

so we use an Gaussian approximation to the conditional

$$\pi(\theta|\boldsymbol{y}) \approx \text{norm.const} \left.\frac{\pi(\boldsymbol{x},\boldsymbol{y}|\theta)\ \pi(\theta)}{\pi_G(\boldsymbol{x}|\boldsymbol{y},\theta)}\right|_{x=\text{xmode}(\theta)}$$

Non-Gaussian observations (I)

In this case

$$\pi(\theta|\boldsymbol{y}) \propto \frac{\overbrace{\pi(\boldsymbol{x},\boldsymbol{y}|\theta)}^{\text{Non-Gaussian, BUT KNOWN}}\ \pi(\theta)}{\underbrace{\pi(\boldsymbol{x}|\boldsymbol{y},\theta)}_{\text{Non-Gaussian and UNKNOWN}}}$$

so we use an Gaussian approximation to the conditional

$$\pi(\theta|\boldsymbol{y}) \approx \text{norm.const} \left.\frac{\pi(\boldsymbol{x},\boldsymbol{y}|\theta)\ \pi(\theta)}{\pi_G(\boldsymbol{x}|\boldsymbol{y},\theta)}\right|_{x=\text{xmode}(\theta)}$$

Non-Gaussian observations (II)

For the marginals for $x_i|\boldsymbol{y},\boldsymbol{\theta}$, we do similarly

$$\pi(x_i|\boldsymbol{y},\boldsymbol{\theta}) \approx \text{norm.const} \left.\frac{\pi(\boldsymbol{x},\boldsymbol{y}|\theta)}{\pi_G(\boldsymbol{x}_{-i}|x_i,\boldsymbol{y},\theta)}\right|_{\boldsymbol{x}_{-i}=\text{xmode}(\theta,x_i)}$$

This is the hard part, as this is potentially very slow.

Non-Gaussian observations (II)

For the marginals for $x_i|\boldsymbol{y},\boldsymbol{\theta}$, we do similarly

$$\pi(x_i|\boldsymbol{y},\boldsymbol{\theta}) \approx \text{norm.const} \frac{\pi(\boldsymbol{x},\boldsymbol{y}|\theta)}{\pi_G(\boldsymbol{x}_{-i}|x_i,\boldsymbol{y},\theta)}\Bigg|_{\boldsymbol{x}_{-i}=\text{xmode}(\theta,x_i)}$$

This is the hard part, as this is potentially very slow.

Main ideas (I)

Main ideas are simple and based on the identity

$$\pi(z) = \frac{\pi(x,z)}{\pi(x|z)} \qquad \text{leading to} \qquad \widetilde{\pi}(z) = \frac{\pi(x,z)}{\widetilde{\pi}(x|z)}$$

When $\widetilde{\pi}(x|z)$ is the Gaussian-approximation, this is the Laplace-approximation.

Main ideas (I)

Main ideas are simple and based on the identity

$$\pi(z) = \frac{\pi(x,z)}{\pi(x|z)} \qquad \text{leading to} \qquad \widetilde{\pi}(z) = \frac{\pi(x,z)}{\widetilde{\pi}(x|z)}$$

When $\widetilde{\pi}(x|z)$ is the Gaussian-approximation, this is the Laplace-approximation.

Main ideas (II)

Construct the approximations to

1. $\pi(\boldsymbol{\theta}|\boldsymbol{y})$
2. $\pi(x_i|\boldsymbol{\theta},\boldsymbol{y})$

then we integrate

$$\pi(x_i|\boldsymbol{y}) = \int \pi(\boldsymbol{\theta}|\boldsymbol{y})\, \pi(x_i|\boldsymbol{\theta},\boldsymbol{y})\, d\boldsymbol{\theta}$$

$$\pi(\theta_j|\boldsymbol{y}) = \int \pi(\boldsymbol{\theta}|\boldsymbol{y})\, d\boldsymbol{\theta}_{-j}$$

Main ideas (II)

Construct the approximations to

1. $\pi(\boldsymbol{\theta}|\boldsymbol{y})$
2. $\pi(x_i|\boldsymbol{\theta},\boldsymbol{y})$

then we integrate

$$\pi(x_i|\boldsymbol{y}) = \int \pi(\boldsymbol{\theta}|\boldsymbol{y})\, \pi(x_i|\boldsymbol{\theta},\boldsymbol{y})\, d\boldsymbol{\theta}$$

$$\pi(\theta_j|\boldsymbol{y}) = \int \pi(\boldsymbol{\theta}|\boldsymbol{y})\, d\boldsymbol{\theta}_{-j}$$

Main ideas (II)

Construct the approximations to

1. $\pi(\boldsymbol{\theta}|\boldsymbol{y})$
2. $\pi(x_i|\boldsymbol{\theta}, \boldsymbol{y})$

then we integrate

$$\pi(x_i|\boldsymbol{y}) = \int \pi(\boldsymbol{\theta}|\boldsymbol{y})\, \pi(x_i|\boldsymbol{\theta}, \boldsymbol{y})\, d\boldsymbol{\theta}$$

$$\pi(\theta_j|\boldsymbol{y}) = \int \pi(\boldsymbol{\theta}|\boldsymbol{y})\, d\boldsymbol{\theta}_{-j}$$

The Gaussian/GMRF-approximation

$$\pi(\boldsymbol{x} \mid \boldsymbol{y}, \boldsymbol{\theta}) \propto \exp\left(-\frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \sum_i \log \pi(y_i|x_i)\right)$$

$$\approx \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T(\boldsymbol{Q} + \text{diag}(c_i))(\boldsymbol{x}-\boldsymbol{\mu})\right) = \pi_G(\boldsymbol{x}|\boldsymbol{\theta}, \boldsymbol{y})$$

Constructed as follows:

- Locate the mode $\boldsymbol{x}^*$
- Expand to second order

Markov properties are preserved!

The Gaussian/GMRF-approximation

$$\pi(\boldsymbol{x} \mid \boldsymbol{y}, \boldsymbol{\theta}) \propto \exp\left(-\frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \sum_i \log \pi(y_i|x_i)\right)$$
$$\approx \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T(\boldsymbol{Q} + \text{diag}(c_i))(\boldsymbol{x}-\boldsymbol{\mu})\right) = \pi_G(\boldsymbol{x}|\boldsymbol{\theta}, \boldsymbol{y})$$

Constructed as follows:

- Locate the mode $\boldsymbol{x}^*$
- Expand to second order

Markov properties are preserved!

The Laplace approximation: The classic case

Compute and approximation to the integral

$$\int \exp(ng(x))\, dx$$

where n is the parameter going to ∞.

Let x_0 be the mode of $g(x)$ and assume $g(x_0) = 0$:

$$g(x) = \frac{1}{2} g''(x_0)(x - x_0)^2 + \cdots .$$

The Laplace approximation: The classic case

Compute and approximation to the integral

$$\int \exp(ng(x))\, dx$$

where n is the parameter going to ∞.

Let x_0 be the mode of $g(x)$ and assume $g(x_0) = 0$:

$$g(x) = \frac{1}{2} g''(x_0)(x - x_0)^2 + \cdots .$$

The Laplace approximation: The classic case...

Then

$$\int \exp(ng(x))\, dx = \sqrt{\frac{2\pi}{n(-g''(x_0))}} + \cdots$$

- As $n \to \infty$, then the integrand gets more and more peaked.
- Error should tends to zero as $n \to \infty$
- Detailed analysis gives

$$\frac{\text{Estimate}(n)}{\text{True}} = 1 + \mathcal{O}(1/n)$$

so the *relative error* is $\mathcal{O}(1/n)$.

The Laplace approximation: The classic case...

Then

$$\int \exp(ng(x))\, dx = \sqrt{\frac{2\pi}{n(-g''(x_0))}} + \cdots$$

- As $n \to \infty$, then the integrand gets more and more peaked.
- Error should tends to zero as $n \to \infty$
- Detailed analysis gives

$$\frac{\text{Estimate}(n)}{\text{True}} = 1 + \mathcal{O}(1/n)$$

so the *relative error* is $\mathcal{O}(1/n)$.

The Laplace approximation: The classic case...

Then

$$\int \exp(ng(x))\, dx = \sqrt{\frac{2\pi}{n(-g''(x_0))}} + \cdots$$

- As $n \to \infty$, then the integrand gets more and more peaked.
- Error should tends to zero as $n \to \infty$
- Detailed analysis gives

$$\frac{\text{Estimate}(n)}{\text{True}} = 1 + \mathcal{O}(1/n)$$

so the *relative error* is $\mathcal{O}(1/n)$.

The Laplace approximation: The classic case...

Then

$$\int \exp(ng(x))\, dx = \sqrt{\frac{2\pi}{n(-g''(x_0))}} + \cdots$$

- As $n \to \infty$, then the integrand gets more and more peaked.
- Error should tends to zero as $n \to \infty$
- Detailed analysis gives

$$\frac{\text{Estimate}(n)}{\text{True}} = 1 + \mathcal{O}(1/n)$$

so the *relative error* is $\mathcal{O}(1/n)$.

Extension I

$$g_n(x) = \frac{1}{n} \sum_{i=1}^{n} g_i(x)$$

then the mode x_0 depends on n as well.

Extension II

$$\int \exp(ng(\boldsymbol{x}))\, d\boldsymbol{x}$$

and $\boldsymbol{x}$ is multivariate, then

$$\int \exp(ng(\boldsymbol{x}))\, d\boldsymbol{x} = \sqrt{\frac{(2\pi)^n}{n|-\boldsymbol{H}|}}$$

where $\boldsymbol{H}$ is the hessian (matrix) at the mode

$$H_{ij} = \frac{\partial^2}{\partial x_i \partial x_j} g(\boldsymbol{x})\Big|_{\boldsymbol{x}=\boldsymbol{x}_0}$$

Extension II

$$\int \exp(ng(\boldsymbol{x}))\, d\boldsymbol{x}$$

and $\boldsymbol{x}$ is multivariate, then

$$\int \exp(ng(\boldsymbol{x}))\, d\boldsymbol{x} = \sqrt{\frac{(2\pi)^n}{n|-\boldsymbol{H}|}}$$

where $\boldsymbol{H}$ is the hessian (matrix) at the mode

$$H_{ij} = \frac{\partial^2}{\partial x_i \partial x_j} g(\boldsymbol{x})\Big|_{\boldsymbol{x}=\boldsymbol{x}_0}$$

Computing marginals

- Our main issue is to compute marginals
- We can use the Laplace-approximation for this issue as well
- A more “statistical” derivation might be appropriate

Computing marginals

- Our main issue is to compute marginals
- We can use the Laplace-approximation for this issue as well
- A more “statistical” derivation might be appropriate

Computing marginals

- Our main issue is to compute marginals
- We can use the Laplace-approximation for this issue as well
- A more “statistical” derivation might be appropriate

Computing marginals...

Consider the general problem

- θ is hyper-parameter with prior $\pi(\theta)$
- x is latent with density $\pi(x|\theta)$
- y is observed with likelihood $\pi(y|x)$

then

$$\pi(\theta|y) = \frac{\pi(x,\theta|y)}{\pi(x|\theta,y)}$$

for any x!

Computing marginals...

Consider the general problem

- θ is hyper-parameter with prior $\pi(\theta)$
- x is latent with density $\pi(x|\theta)$
- y is observed with likelihood $\pi(y|x)$

then

$$\pi(\theta|y) = \frac{\pi(x,\theta|y)}{\pi(x|\theta,y)}$$

for any x!

Computing marginals...

Consider the general problem

- θ is hyper-parameter with prior $\pi(\theta)$
- x is latent with density $\pi(x|\theta)$
- y is observed with likelihood $\pi(y|x)$

then

$$\pi(\theta|y) = \frac{\pi(x,\theta|y)}{\pi(x|\theta,y)}$$

for any x!

Computing marginals...

Some details

$$\begin{aligned}
\pi(\theta|y) &= \frac{\pi(x,\theta|y)}{\pi(x|\theta,y)} \\
&\propto \frac{\pi(\theta)\,\pi(x|\theta)\,\pi(y|x)}{\pi(x|\theta,y)} \\
&\approx \left.\frac{\pi(\theta)\,\pi(x|\theta)\,\pi(y|x)}{\pi_G(x|\theta,y)}\right|_{x=x^*(\theta)}
\end{aligned}$$

where $\pi_G(x|\theta,y)$ is the Gaussian approximation of $\pi(x|\theta,y)$ and $x^*(\theta)$ is the mode.

Errors in the approximations

Result:[3]

With n repeated measurements $\boldsymbol{y}$ *of the same* $\boldsymbol{x}$*, then*

$$\frac{\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{y}_n)}{\pi(\boldsymbol{\theta}|\boldsymbol{y}_n)} = 1 + \mathcal{O}(n^{-3/2})$$

after renormalisation.

Relative error is a very very very nice property!

The error-rate is impressive!

Unfortunately, the assumptions made are not usually valid...

[3] Tierney & Kadane, JASA, 1986

Errors in the approximations

Result:[3]

With n repeated measurements $\boldsymbol{y}$ *of the same* $\boldsymbol{x}$*, then*

$$\frac{\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{y}_n)}{\pi(\boldsymbol{\theta}|\boldsymbol{y}_n)} = 1 + \mathcal{O}(n^{-3/2})$$

after renormalisation.

Relative error is a very very very nice property!

The error-rate is impressive!

Unfortunately, the assumptions made are not usually valid...

[3] Tierney & Kadane, JASA, 1986

Errors in the approximations

Result:[3]

With n repeated measurements $\boldsymbol{y}$ *of the same* $\boldsymbol{x}$*, then*

$$\frac{\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{y}_n)}{\pi(\boldsymbol{\theta}|\boldsymbol{y}_n)} = 1 + \mathcal{O}(n^{-3/2})$$

after renormalisation.

Relative error is a very very very nice property!

The error-rate is impressive!

Unfortunately, the assumptions made are not usually valid...

[3] Tierney & Kadane, JASA, 1986

Errors in the approximations

Result:[3]

With n repeated measurements $\boldsymbol{y}$ *of the same* $\boldsymbol{x}$*, then*

$$\frac{\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{y}_n)}{\pi(\boldsymbol{\theta}|\boldsymbol{y}_n)} = 1 + \mathcal{O}(n^{-3/2})$$

after renormalisation.

Relative error is a very very very nice property!

The error-rate is impressive!

Unfortunately, the assumptions made are not usually valid...

[3] Tierney & Kadane, JASA, 1986

Summary

- This are the basic ideas
- The rest are just details, but there are a lot of them...

Summary

- This are the basic ideas
- The rest are just details, but there are a lot of them...

Extensions

- Model choice/selection
- Automatic detection of “surprising” observations
- High(er) number of hyperparameters

Extensions

- Model choice/selection
- Automatic detection of “surprising” observations
- High(er) number of hyperparameters

Model choice

Chose/compare various model is important but difficult

- Bayes factors (general available)
- Deviance information criterion (DIC) (hierarchical models)

Marginal likelihood

Marginal likelihood is the normalising constant for $\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{y})$,

$$\widetilde{\pi}(\boldsymbol{y}) = \int \left. \frac{\pi(\boldsymbol{\theta})\pi(\boldsymbol{x}|\theta)\pi(\boldsymbol{y}|\boldsymbol{x},\theta)}{\widetilde{\pi}_{\mathsf{G}}(\boldsymbol{x}|\boldsymbol{\theta},\boldsymbol{y})} \right|_{\boldsymbol{x}=\boldsymbol{x}^\star(\boldsymbol{\theta})} d\boldsymbol{\theta}. \tag{7}$$

I many hierarchical GMRF models the prior is intrinsic/improper, so this is difficult to use.

Marginal likelihood

Marginal likelihood is the normalising constant for $\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{y})$,

$$\widetilde{\pi}(\boldsymbol{y}) = \int \left. \frac{\pi(\boldsymbol{\theta})\pi(\boldsymbol{x}|\theta)\pi(\boldsymbol{y}|\boldsymbol{x},\theta)}{\widetilde{\pi}_{\mathsf{G}}(\boldsymbol{x}|\boldsymbol{\theta},\boldsymbol{y})} \right|_{\boldsymbol{x}=\boldsymbol{x}^\star(\boldsymbol{\theta})} d\boldsymbol{\theta}. \tag{7}$$

I many hierarchical GMRF models the prior is intrinsic/improper, so this is difficult to use.

Deviance Information Criteria

Based on the *deviance*

$$D(\boldsymbol{x};\boldsymbol{\theta}) = -2\sum_i \log(y_i \mid x_i,\boldsymbol{\theta})$$

and

$$DIC = 2 \times \text{Mean}\left(D(\boldsymbol{x};\boldsymbol{\theta})\right) - D(\text{Mean}(\boldsymbol{x});\boldsymbol{\theta}^*)$$

This is quite easy to compute

Bayesian Cross-validation

Easy to compute using the INLA-approach

$$\pi(y_i \mid \boldsymbol{y}_{-i}) = \int_{\boldsymbol{\theta}} \left\{ \int_{x_i} \pi(y_i \mid x_i, \boldsymbol{\theta})\, \pi(x_i \mid \boldsymbol{y}_{-i}, \boldsymbol{\theta})\, dx_i \right\} \pi(\boldsymbol{\theta} \mid \boldsymbol{y}_{-i})\, d\boldsymbol{\theta}$$

where

$$\pi(x_i \mid \boldsymbol{y}_{-i}, \boldsymbol{\theta}) \propto \frac{\pi(x_i|\boldsymbol{y}, \boldsymbol{\theta})}{\pi(y_i|x_i, \boldsymbol{\theta})}$$

Require a one-dimensional integral for each i and $\boldsymbol{\theta}$.

Automatic detection of “surprising” observations

Compute

$$\text{Prob}(y_i^{\text{new}} \leq y_i \mid \boldsymbol{y}_{-i})$$

Look for unusual large or small values

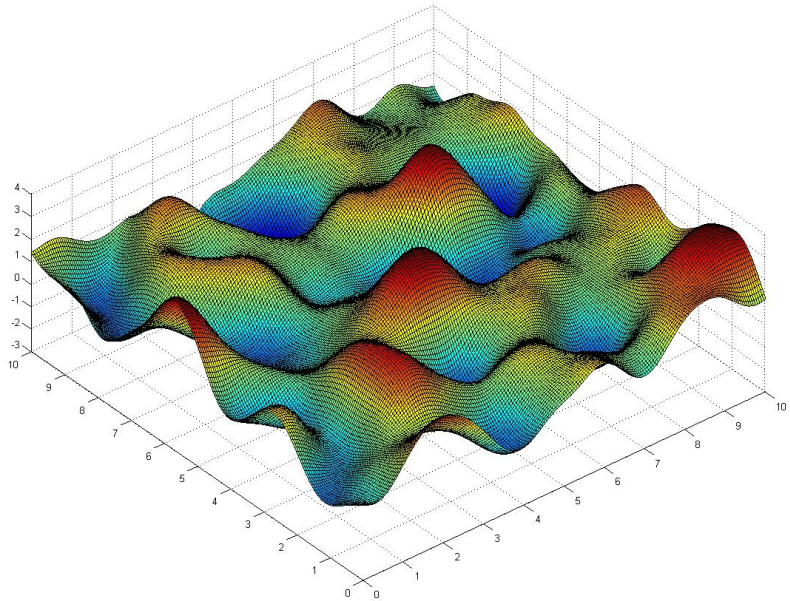
Part VII

Gaussian fields and GMRFs

Gaussian fields

Gaussian fields is central in spatial statistics!

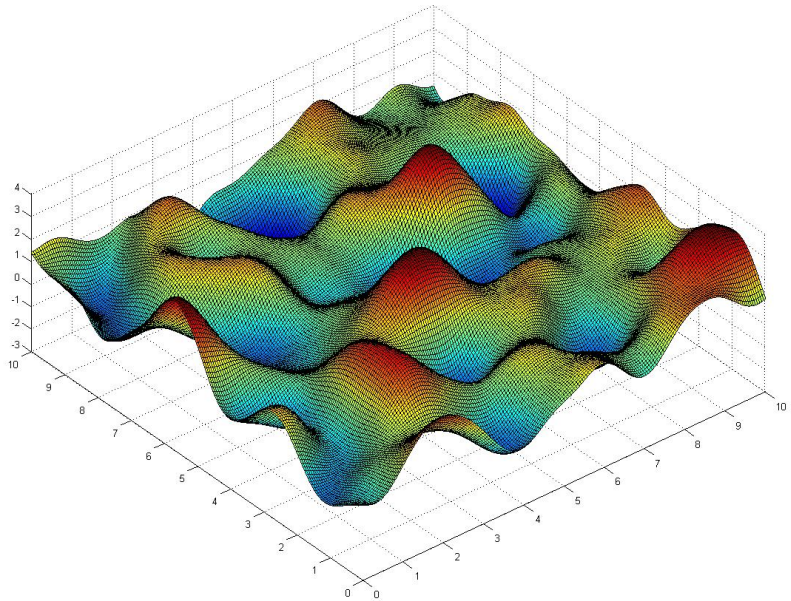
- Covariance function, often depends only on distances
- Matérn family
- Dense covariance matrix
- Known marginal properties
- No boundary issues
- Resolution consistent
- Computational properties are not good: $\mathcal{O}(n^3)$
- Difficulties: non-stationarity, space-time and curved spaces



Gaussian fields

Gaussian fields is central in spatial statistics!

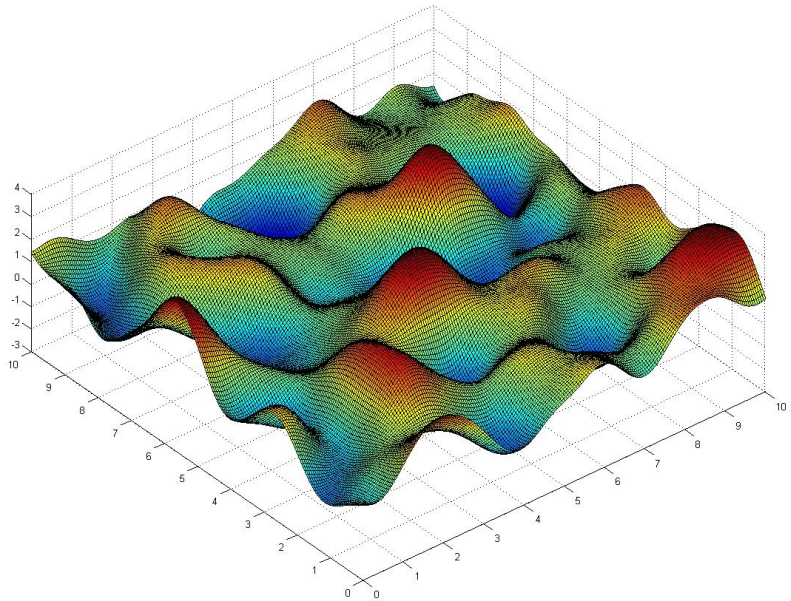
- Covariance function, often depends only on distances
- Matérn family
- Dense covariance matrix
- Known marginal properties
- No boundary issues
- Resolution consistent
- Computational properties are not good: $\mathcal{O}(n^3)$
- Difficulties: non-stationarity, space-time and curved spaces



Gaussian fields

Gaussian fields is central in spatial statistics!

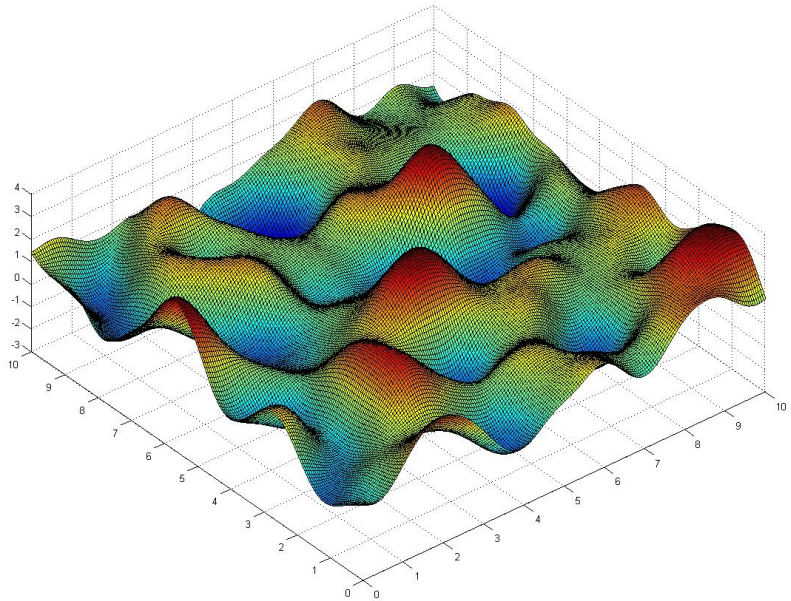
- Covariance function, often depends only on distances
- Matérn family
- Dense covariance matrix
- Known marginal properties
- No boundary issues
- Resolution consistent
- Computational properties are not good: $\mathcal{O}(n^3)$
- Difficulties: non-stationarity, space-time and curved spaces



Gaussian fields

Gaussian fields is central in spatial statistics!

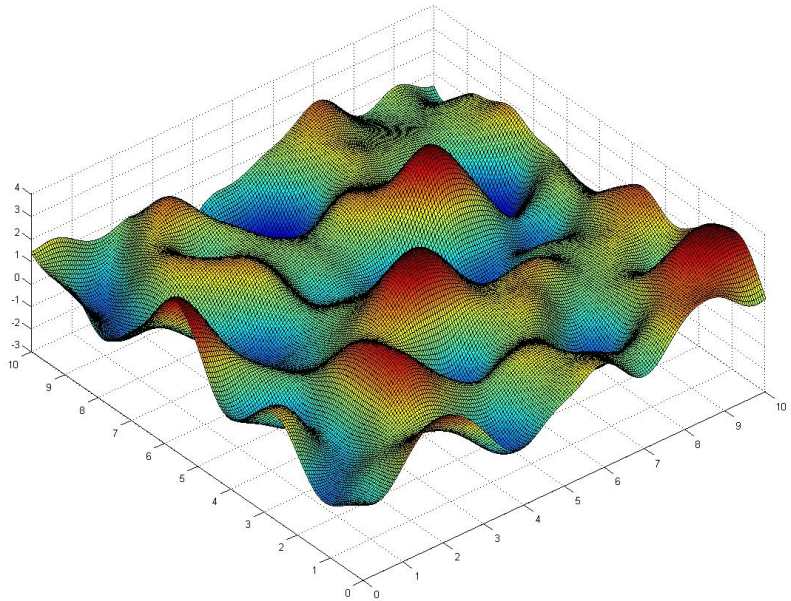
- Covariance function, often depends only on distances
- Matérn family
- Dense covariance matrix
- Known marginal properties
- No boundary issues
- Resolution consistent
- Computational properties are not good: $\mathcal{O}(n^3)$
- Difficulties: non-stationarity, space-time and curved spaces



Gaussian fields

Gaussian fields is central in spatial statistics!

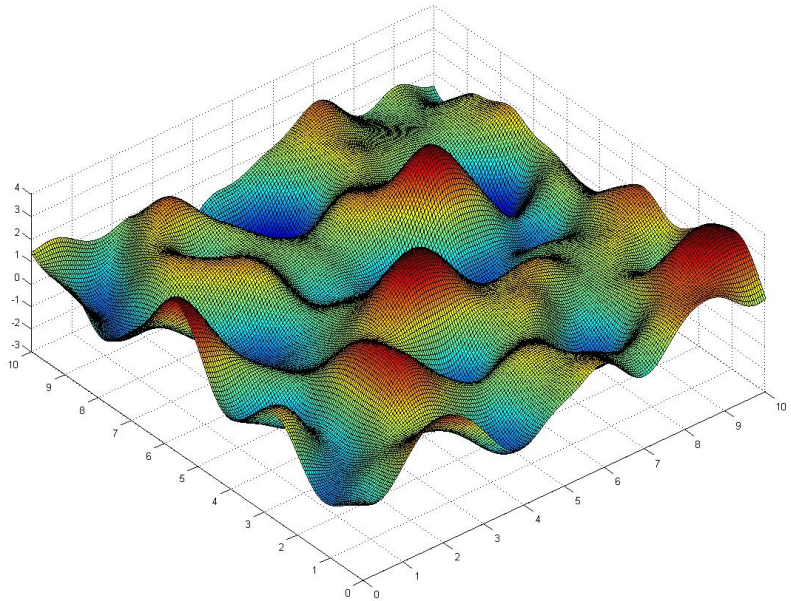
- Covariance function, often depends only on distances
- Matérn family
- Dense covariance matrix
- Known marginal properties
- No boundary issues
- Resolution consistent
- Computational properties are not good: $\mathcal{O}(n^3)$
- Difficulties: non-stationarity, space-time and curved spaces



Gaussian fields

Gaussian fields is central in spatial statistics!

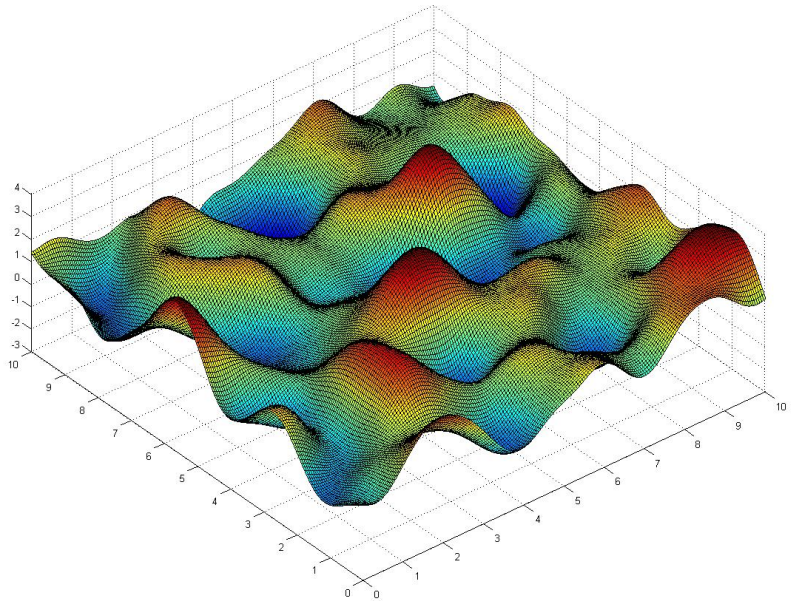
- Covariance function, often depends only on distances
- Matérn family
- Dense covariance matrix
- Known marginal properties
- No boundary issues
- Resolution consistent
- Computational properties are not good: $\mathcal{O}(n^3)$
- Difficulties: non-stationarity, space-time and curved spaces



Gaussian fields

Gaussian fields is central in spatial statistics!

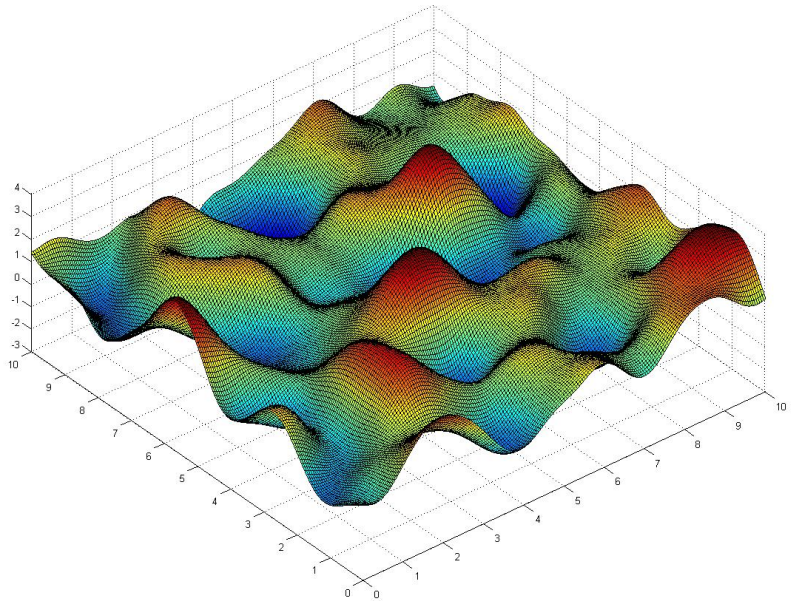
- Covariance function, often depends only on distances
- Matérn family
- Dense covariance matrix
- Known marginal properties
- No boundary issues
- Resolution consistent
- Computational properties are not good: $\mathcal{O}(n^3)$
- Difficulties: non-stationarity, space-time and curved spaces



Gaussian fields

Gaussian fields is central in spatial statistics!

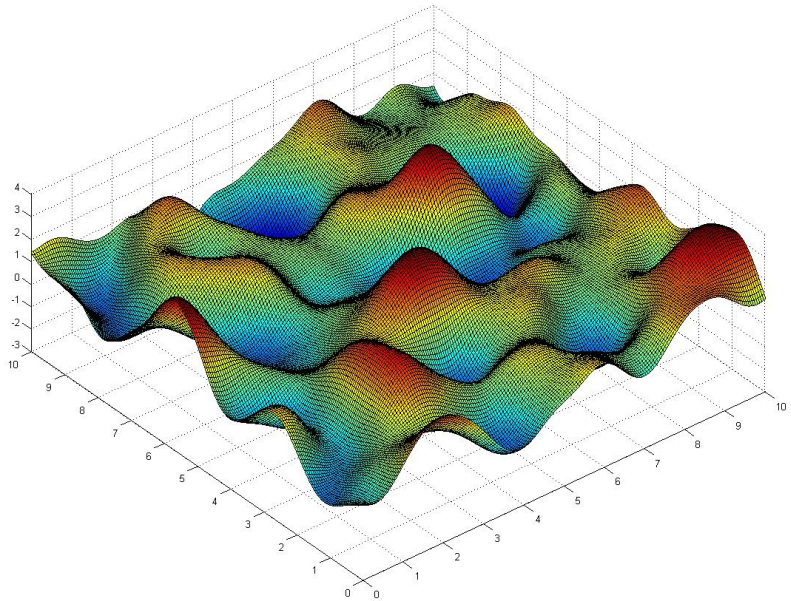
- Covariance function, often depends only on distances
- Matérn family
- Dense covariance matrix
- Known marginal properties
- No boundary issues
- Resolution consistent
- Computational properties are not good: $\mathcal{O}(n^3)$
- Difficulties: non-stationarity, space-time and curved spaces



Gaussian fields

Gaussian fields is central in spatial statistics!

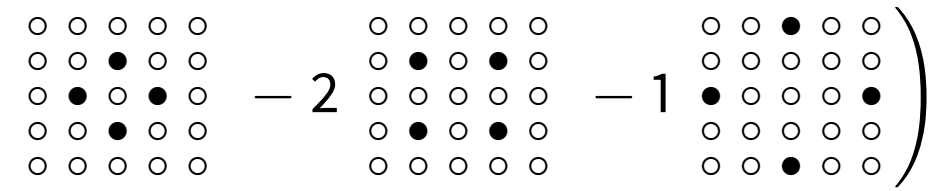
- Covariance function, often depends only on distances
- Matérn family
- Dense covariance matrix
- Known marginal properties
- No boundary issues
- Resolution consistent
- Computational properties are not good: $\mathcal{O}(n^3)$
- Difficulties: non-stationarity, space-time and curved spaces



Full conditionals

- $x_i | \boldsymbol{x}_{-i}$ only depends on the (few nearest) neighbours

$$\mathrm{E}(x_{ij} \mid \boldsymbol{x}_{-ij}) = \frac{1}{20}\left(8 \; - 2 \; - 1 \; \right)$$



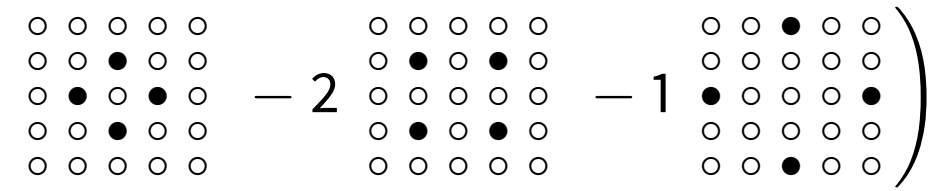
$$\mathrm{Prec}(x_{ij} \mid \boldsymbol{x}_{-ij}) = 20\kappa.$$

- Simple conditional interpretation
- Small memory footprint
- Fast computations: $\mathcal{O}(n^{3/2})$ in $\mathbb{R}^2$

Full conditionals

- $x_i | \boldsymbol{x}_{-i}$ only depends on the (few nearest) neighbours

$$\mathrm{E}(x_{ij} \mid \boldsymbol{x}_{-ij}) = \frac{1}{20}\left(8 \; - 2 \; - 1 \; \right)$$



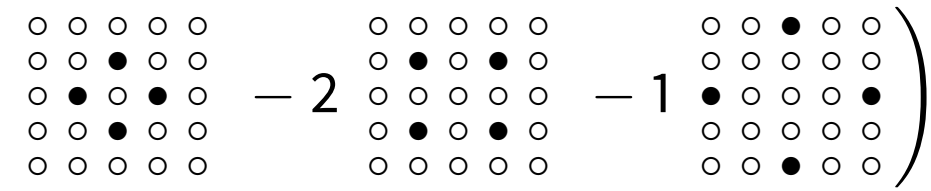
$$\mathrm{Prec}(x_{ij} \mid \boldsymbol{x}_{-ij}) = 20\kappa.$$

- Simple conditional interpretation
- Small memory footprint
- Fast computations: $\mathcal{O}(n^{3/2})$ in $\mathbb{R}^2$

Full conditionals

- $x_i | \boldsymbol{x}_{-i}$ only depends on the (few nearest) neighbours

$$\mathrm{E}(x_{ij} \mid \boldsymbol{x}_{-ij}) = \frac{1}{20}\left(8 \; - 2 \; - 1 \; \right)$$



$$\mathrm{Prec}(x_{ij} \mid \boldsymbol{x}_{-ij}) = 20\kappa.$$

- Simple conditional interpretation
- Small memory footprint
- Fast computations: $\mathcal{O}(n^{3/2})$ in $\mathbb{R}^2$

Full conditionals

- $x_i | \boldsymbol{x}_{-i}$ only depends on the (few nearest) neighbours

$$\mathrm{E}(x_{ij} \mid \boldsymbol{x}_{-ij}) = \frac{1}{20}\left(8 \; - 2 \; - 1 \; \right)$$

$$\mathrm{Prec}(x_{ij} \mid \boldsymbol{x}_{-ij}) = 20\kappa.$$

- Simple conditional interpretation
- Small memory footprint
- Fast computations: $\mathcal{O}(n^{3/2})$ in $\mathbb{R}^2$

Why should we care about GMRFs?

- Conditional modeling
- Strongest argument: Computational speed!
- Sparse matrices allow for faster computation, general algorithms
- Also relevant for sampling based inference

Why should we care about GMRFs?

- Conditional modeling
- Strongest argument: Computational speed!
- Sparse matrices allow for faster computation, general algorithms
- Also relevant for sampling based inference

Why should we care about GMRFs?

- Conditional modeling
- Strongest argument: Computational speed!
- Sparse matrices allow for faster computation, general algorithms
- Also relevant for sampling based inference

Why should we care about GMRFs?

- Conditional modeling
- Strongest argument: Computational speed!
- Sparse matrices allow for faster computation, general algorithms
- Also relevant for sampling based inference

The downside...

- Not easy to specify GMRFs to get "what you want"
- The link from

$$\{\pi(x_i|\boldsymbol{x}_{-i})\}$$

to

$$\mathrm{Cov}(x_i, x_i)$$

is "difficult" (without computing it)
- Boundary issues
- Irregular lattices are even more difficult
- Commonly used GMRF models are not resolution consistent!

The downside...

- Not easy to specify GMRFs to get "what you want"
- The link from

$$\{\pi(x_i|\boldsymbol{x}_{-i})\}$$

to

$$\text{Cov}(x_i, x_i)$$

is "difficult" (without computing it)

- Boundary issues
- Irregular lattices are even more difficult
- Commonly used GMRF models are not resolution consistent!

The downside...

- Not easy to specify GMRFs to get "what you want"
- The link from

$$\{\pi(x_i|\boldsymbol{x}_{-i})\}$$

to

$$\text{Cov}(x_i, x_i)$$

is "difficult" (without computing it)

- Boundary issues
- Irregular lattices are even more difficult
- Commonly used GMRF models are not resolution consistent!

The downside...

- Not easy to specify GMRFs to get "what you want"
- The link from

$$\{\pi(x_i|\boldsymbol{x}_{-i})\}$$

to

$$\text{Cov}(x_i, x_i)$$

is "difficult" (without computing it)

- Boundary issues
- Irregular lattices are even more difficult
- Commonly used GMRF models are not resolution consistent!

The downside...

- Not easy to specify GMRFs to get "what you want"
- The link from

$$\{\pi(x_i|\boldsymbol{x}_{-i})\}$$

to

$$\text{Cov}(x_i, x_i)$$

is "difficult" (without computing it)

- Boundary issues
- Irregular lattices are even more difficult
- Commonly used GMRF models are not resolution consistent!

Matérn fields and the reference SPDE

The solution of this SPDE (Whittle 1954/1963)

$$(\kappa^2 - \Delta)^{\alpha/2} \boldsymbol{x}(\boldsymbol{s}) = \boldsymbol{\epsilon}(\boldsymbol{s}), \qquad \alpha = \nu + \text{dim}/2$$

is a Gaussian field with Matérn Covariance function. Δ is the Laplacian and $\boldsymbol{\epsilon}(\boldsymbol{s})$ is spatial Gaussian white noise.

The SPDE *is* harder to work with, *but* when we know *how* then

- manifolds (easy!)
- non-stationary (easy!)
- computational properties (very good!)
- +++

Matérn fields and the reference SPDE

The solution of this SPDE (Whittle 1954/1963)

$$(\kappa^2 - \Delta)^{\alpha/2} \boldsymbol{x}(\boldsymbol{s}) = \boldsymbol{\epsilon}(\boldsymbol{s}), \qquad \alpha = \nu + \text{dim}/2$$

is a Gaussian field with Matérn Covariance function. Δ is the Laplacian and $\boldsymbol{\epsilon}(\boldsymbol{s})$ is spatial Gaussian white noise.

The SPDE *is* harder to work with, *but* when we know *how* then

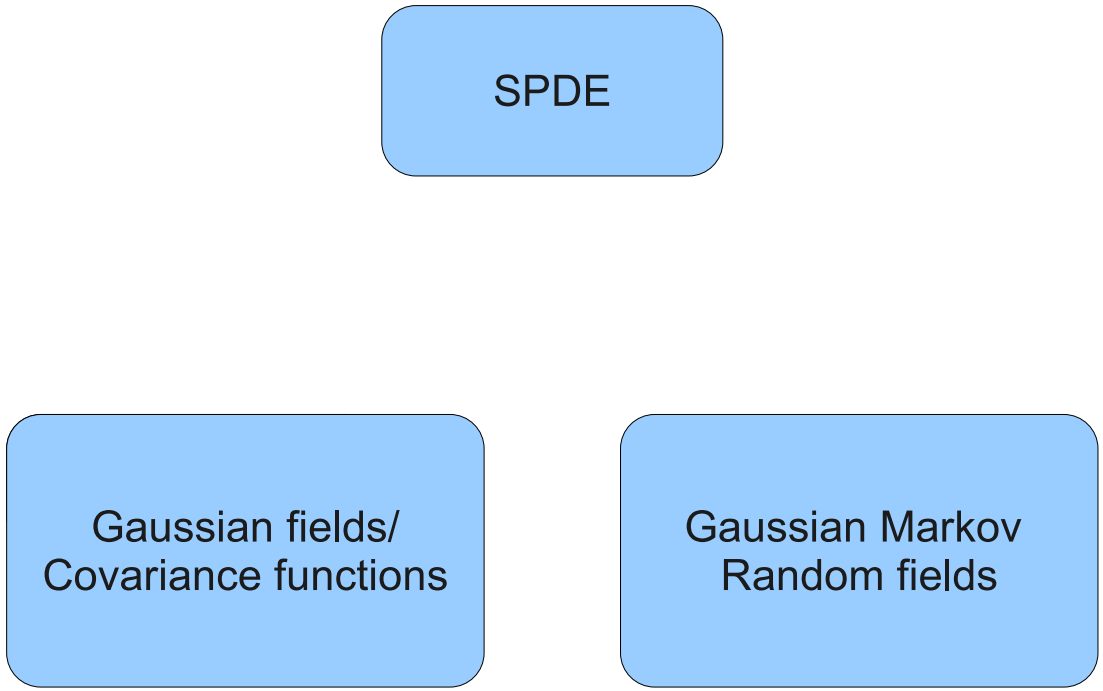
- manifolds (easy!)
- non-stationary (easy!)
- computational properties (very good!)
- +++

Main result

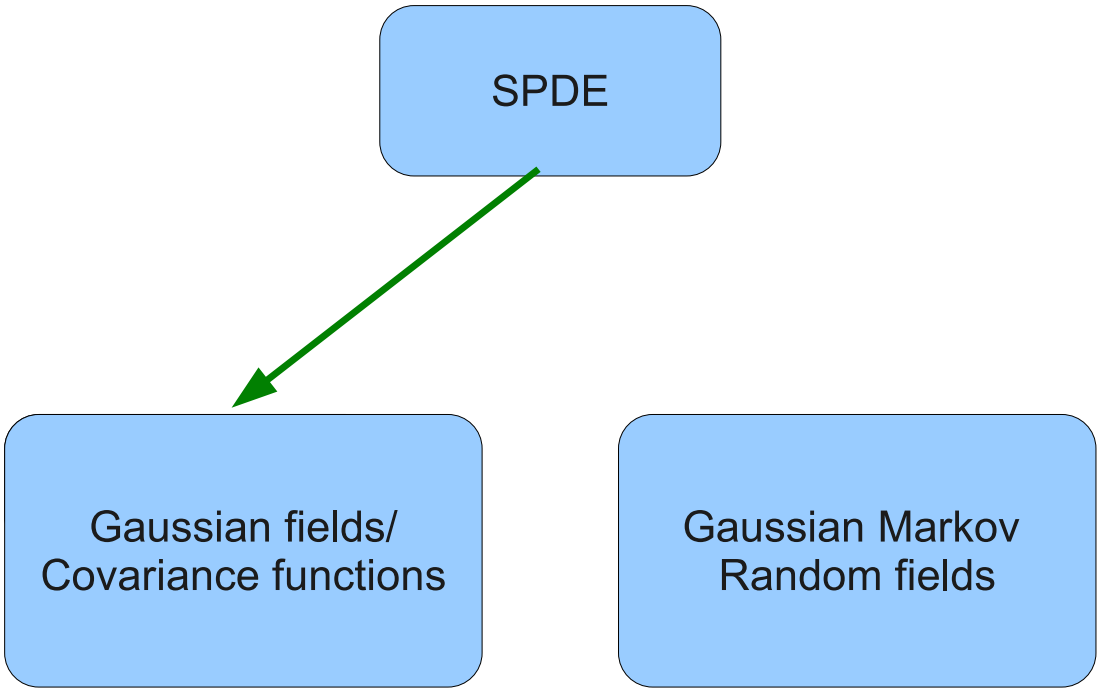
Gaussian fields/
Covariance functions

Gaussian Markov
Random fields

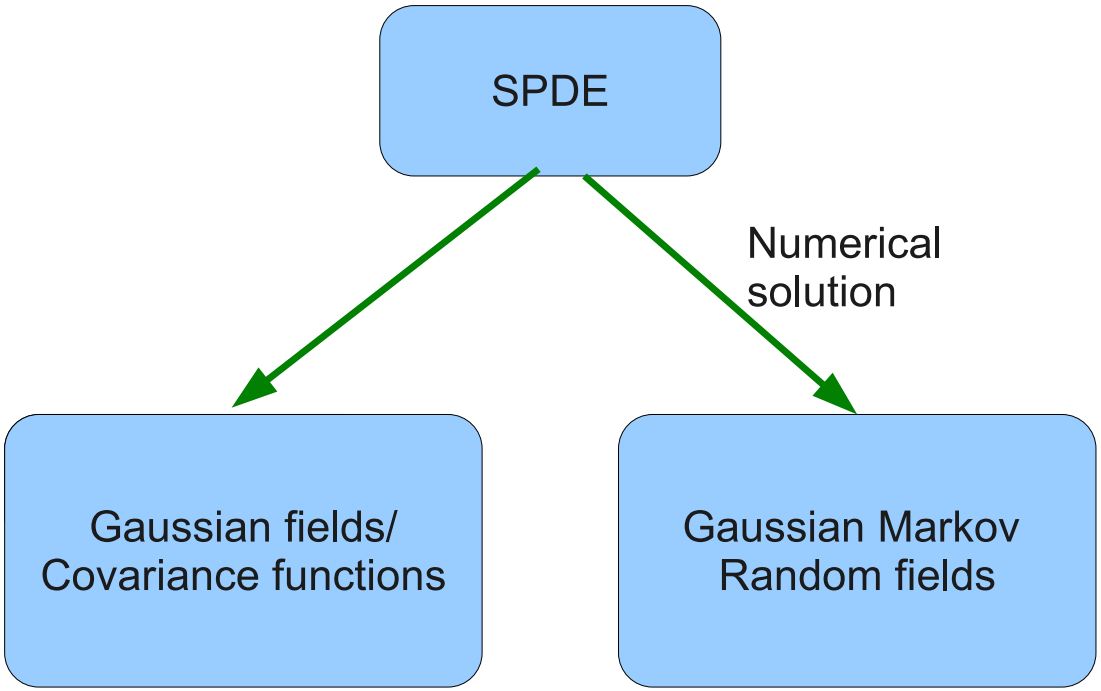
Main result


SPDE
Gaussian fields/
Covariance functions
Gaussian Markov
Random fields

Main result


SPDE
Gaussian fields/
Covariance functions
Gaussian Markov
Random fields

Main result


SPDE
Numerical solution
Gaussian fields/ Covariance functions
Gaussian Markov Random fields

Main result

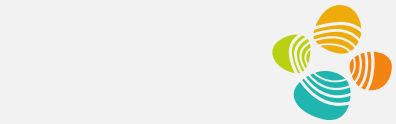
1. A numerical solution to the SPDE can be constructed as

$$x(s) = \sum_{i=1}^{n} w_i \phi_i(s)$$

for local basis/"tent" functions $\{\phi_i(s)\}$ where the weights is a local GMRF with precision matrix $\boldsymbol{Q}(\cdot)$, for $\alpha = 1, 2, 3, \ldots$.

2. We can construct $\boldsymbol{Q}$ from a triangulation at no cost.

Main result



1. A numerical solution to the SPDE can be constructed as

$$x(s) = \sum_{i=1}^{n} w_i \phi_i(s)$$

for local basis/"tent" functions $\{\phi_i(s)\}$ where the weights is a local GMRF with precision matrix $\boldsymbol{Q}(\cdot)$, for $\alpha = 1, 2, 3, \ldots$.

2. We can construct $\boldsymbol{Q}$ from a triangulation at no cost.

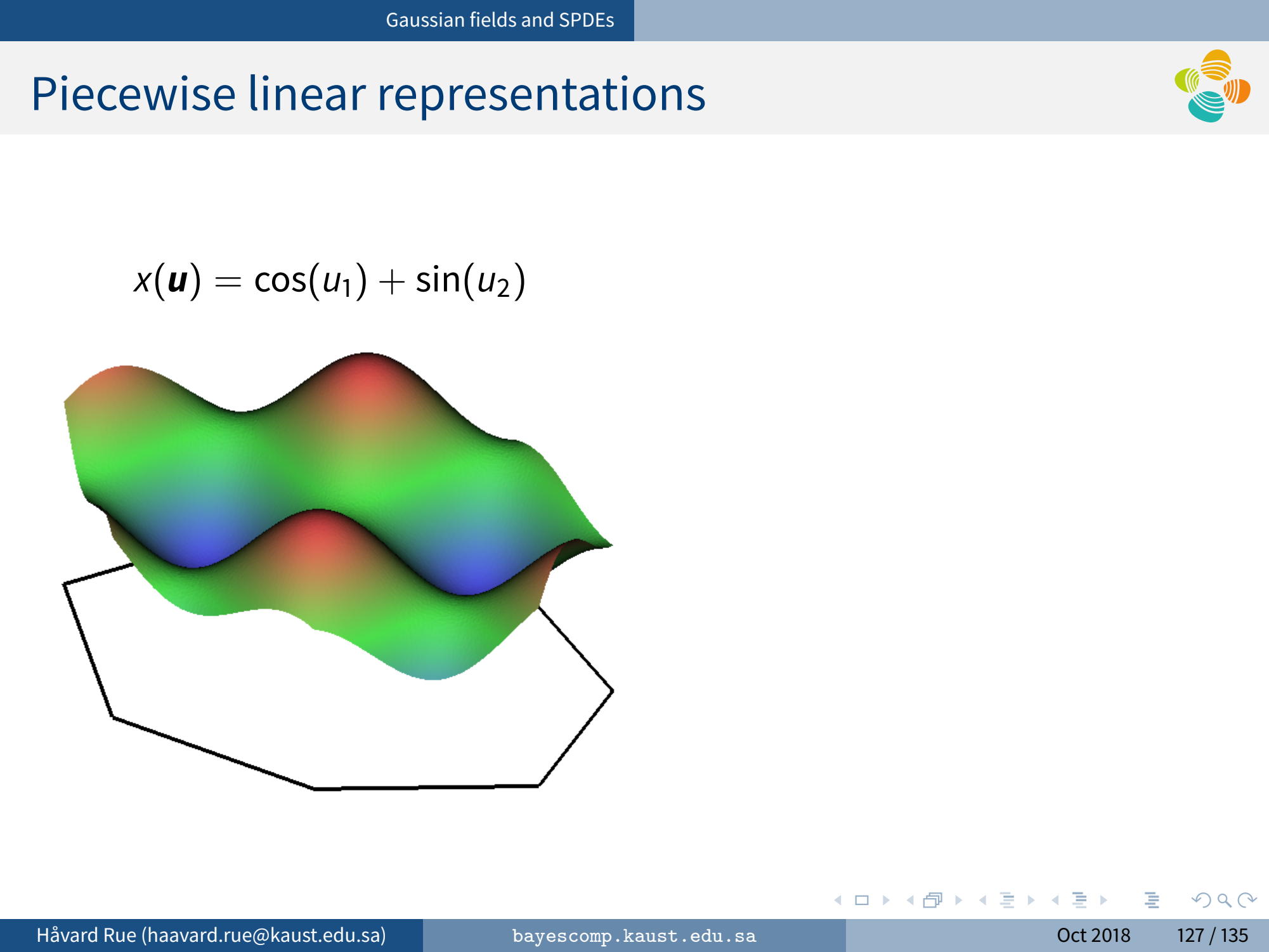
How to "work" with the SPDE?

Hilbert space representation/finite element method

$$\boldsymbol{x}(\boldsymbol{u}) = \sum_{k=1}^{N} \psi_k(\boldsymbol{u}) w_k$$

for basis-functions $\{\psi_k\}$ and (Gaussian) weights $\{w_k\}$

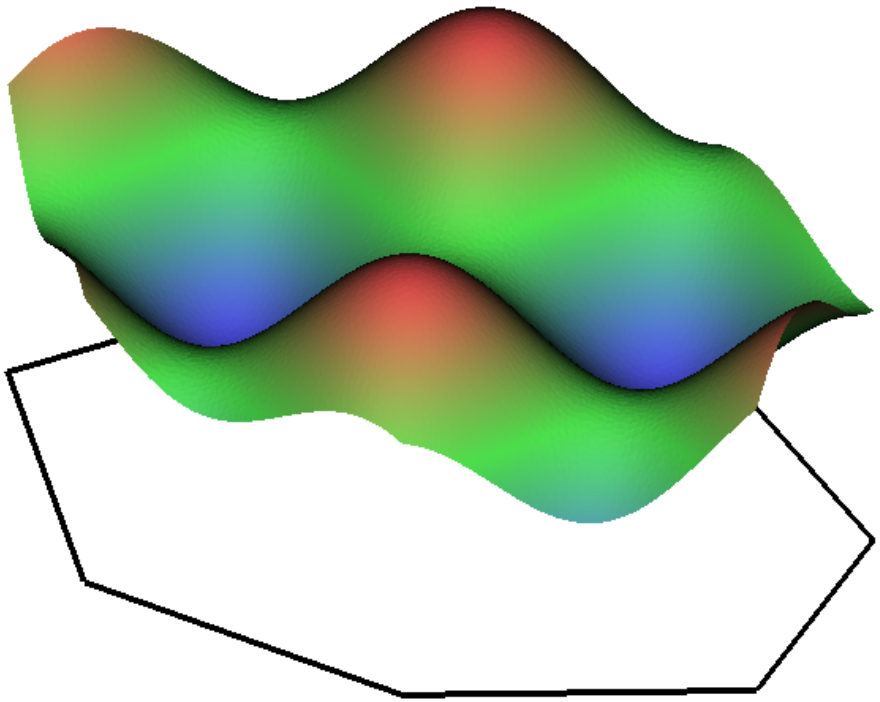
Piecewise linear representations



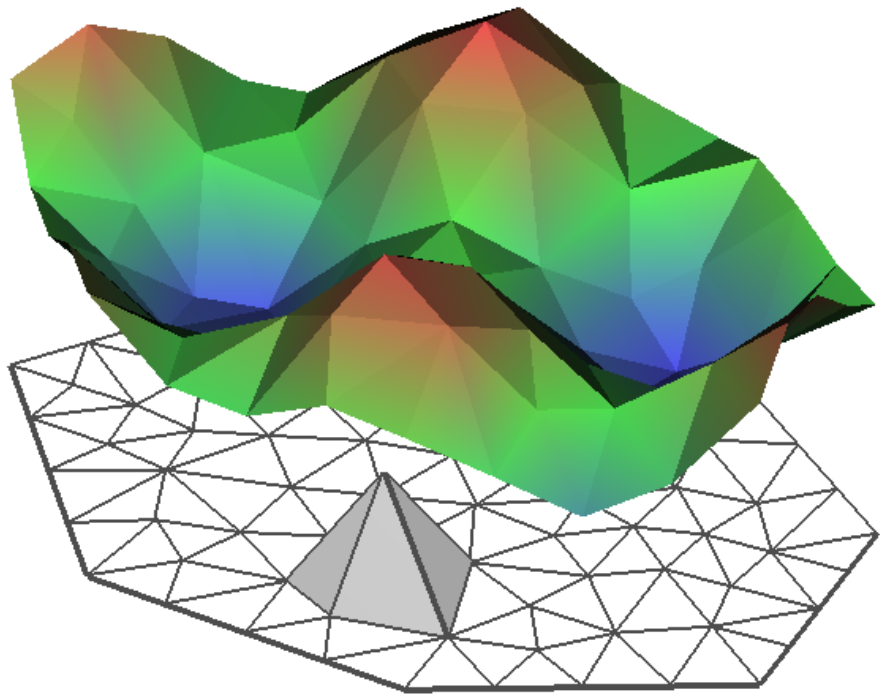
$$x(\boldsymbol{u}) = \cos(u_1) + \sin(u_2)$$

Piecewise linear representations

$$x(\boldsymbol{u}) = \cos(u_1) + \sin(u_2)$$



$$x(\boldsymbol{u}) = \sum_k \psi_k(\boldsymbol{u})\, x_k$$



(Stochastic) Weak solution

$$(\kappa^2 - \Delta)^{\alpha/2} \boldsymbol{x}(\boldsymbol{s}) = \boldsymbol{\epsilon}(\boldsymbol{s})$$

(Stochastic) Weak solution

$$\left\{ \langle \phi_k, (\kappa^2 - \Delta)^{\alpha/2} \boldsymbol{x} \rangle \right\}_k \stackrel{D}{=} \{ \langle \phi_k, \boldsymbol{\epsilon} \rangle \}_k$$

for all *test functions* $\{\phi_k\}_k$.

$$\left(\quad \langle f, g \rangle \stackrel{\text{def}}{=} \int f(\boldsymbol{s}) g(\boldsymbol{s}) \, d\boldsymbol{s} \quad \right)$$

Results for $\alpha = 1, 2, \ldots$ (I)

$$\alpha = 1: \qquad \phi_k = (\kappa^2 - \Delta)^{1/2} \psi_k$$
$$\alpha = 2: \qquad \phi_k = \psi_k$$

Define matrices $\boldsymbol{C}$, $\boldsymbol{G}$ and $\boldsymbol{K}$

$$C_{ij} = \langle \phi_i, \phi_j \rangle \quad i \neq j$$
$$G_{ij} = \langle \nabla \phi_i, \nabla \phi_j \rangle$$
$$\boldsymbol{K} = \kappa^2 \boldsymbol{C} + \boldsymbol{G}$$

Results for $\alpha = 1, 2, \ldots$ (II)

The weights are Gaussian with precision matrix $\boldsymbol{Q}_{\alpha,\kappa}$

$$\begin{aligned}
\boldsymbol{Q}_{1,\kappa} &= \kappa^2 \boldsymbol{C} + \boldsymbol{G} \\
\boldsymbol{Q}_{2,\kappa} &= \boldsymbol{K}\boldsymbol{C}^{-1}\boldsymbol{K} \\
&\vdots \\
\boldsymbol{Q}_{\alpha,\kappa} &= \boldsymbol{K}\boldsymbol{C}^{-1}\boldsymbol{Q}_{\alpha-2,\kappa}\boldsymbol{C}^{-1}\boldsymbol{K}
\end{aligned}$$

Replace $\boldsymbol{C}$ with a appropriate diagonal matrix: This is OK

Result

Can "solve"

$$(\kappa^2 - \Delta)^{\alpha/2} \boldsymbol{x}(\boldsymbol{s}) = \boldsymbol{\epsilon}(\boldsymbol{s}), \quad \alpha = 1, 2, 3, \ldots, \quad \alpha = \nu + \text{dim}/2$$

for

- any κ
- any triangulation

at no cost!!!

"solve" means: we can write down the corresponding precision matrix for the weights, and the precision matrix is (very-)sparse/a-GMRF.

Result

Can "solve"

$$(\kappa^2 - \Delta)^{\alpha/2} \boldsymbol{x}(\boldsymbol{s}) = \boldsymbol{\epsilon}(\boldsymbol{s}), \quad \alpha = 1, 2, 3, \ldots, \quad \alpha = \nu + \text{dim}/2$$

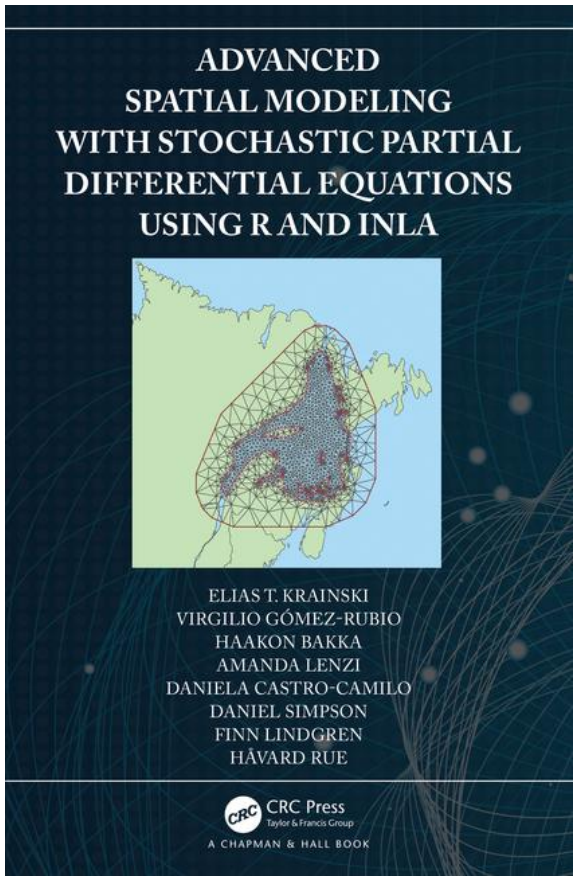
for

- any κ
- any triangulation

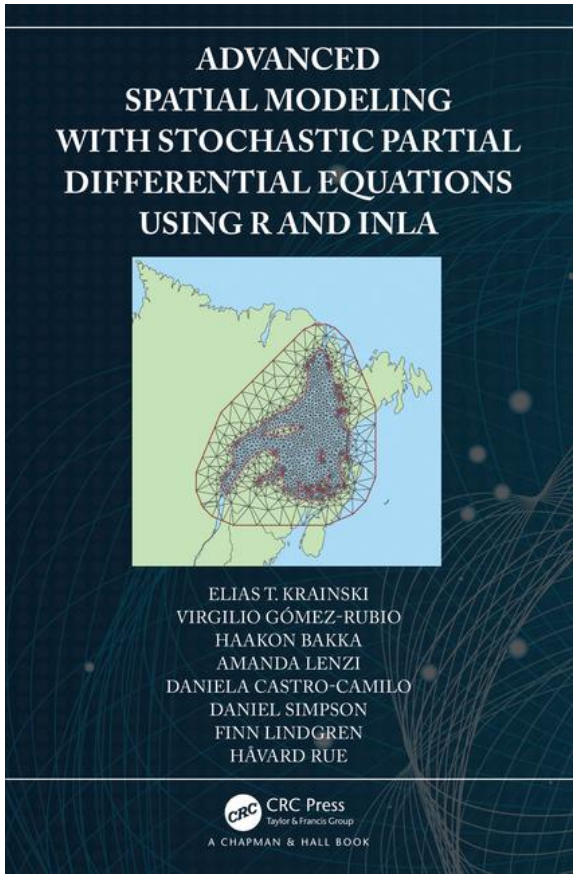
at no cost!!!

"solve" means: we can write down the corresponding precision matrix for the weights, and the precision matrix is (very-)sparse/a-GMRF.

This was just the beginning...



This was just the beginning...



Part VIII

Learn more?

Two recent review papers: part I and II

Bayesian Computing with INLA: A Review

Håvard Rue,[1] Andrea Riebler,[1] Sigrunn H. Sørbye,[2] Janine B. Illian,[3] Daniel P. Simpson,[4] and Finn K. Lindgren[5]

[1]Department of Mathematical Sciences, Norwegian University of Science and Technology, N-7491 Trondheim, Norway; email: hrue@math.ntnu.no

[2]Department of Mathematics and Statistics, The Arctic University of Norway, 9037 Tromsø, Norway

[3]Centre for Research into Ecological and Environmental Modelling, School of Mathematics and Statistics, University of St. Andrews, KY16 9LZ Fife, United Kingdom

[4]Department of Mathematical Sciences, University of Bath, BA2 7AY Bath, United Kingdom

[5]School of Mathematics, The University of Edinburgh, EH9 3FD Edinburgh, United Kingdom

Annu. Rev. Stat. Appl. 2017. 4:395–421

First published online as a Review in Advance on December 23, 2016

The *Annual Review of Statistics and Its Application* is online at statistics.annualreviews.org

This article's doi: 10.1146/annurev-statistics-060116-054045

Keywords

Gaussian Markov random fields, Laplace approximations, approximate Bayesian inference, latent Gaussian models, numerical integration, sparse matrices

Abstract

The key operation in Bayesian inference is to compute high-dimensional integrals. An old approximate technique is the Laplace method or approximation, which dates back to Pierre-Simon Laplace (1774). This simple idea

Two recent review papers: part I and II

Bayesian Computing with INLA: A Review

Håvard Rue,[1] Andrea Riebler,[1] Sigrunn H. Sørbye,[2] Janine B. Illian,[3] Daniel P. Simpson,[4] and Finn K. Lindgren[5]

[1]Department of Mathematical Sciences, Norwegian University of Science and Technology, N-7491 Trondheim, Norway; email: hrue@math.ntnu.no

[2]Department of Mathematics and Statistics, The Arctic University of Norway, 9037 Tromsø, Norway

[3]Centre for Research into Ecological and Environmental Modelling, School of Mathematics and Statistics, University of St. Andrews, KY16 9LZ Fife, United Kingdom

[4]Department of Mathematical Sciences, University of Bath, BA2 7AY Bath, United Kingdom

[5]School of Mathematics, The University of Edinburgh, EH9 3FD Edinburgh, United Kingdom

Annu. Rev. Stat. Appl. 2017. 4:395–421

First published online as a Review in Advance on December 23, 2016

The *Annual Review of Statistics and Its Application* is online at statistics.annualreviews.org

This article's doi: 10.1146/annurev-statistics-060116-054045

Keywords

Gaussian Markov random fields, Laplace approximations, approximate Bayesian inference, latent Gaussian models, numerical integration, sparse matrices

Abstract

The key operation in Bayesian inference is to compute high-dimensional integrals. An old approximate technique is the Laplace method or approximation, which dates back to Pierre-Simon Laplace (1774). This simple idea

Received: 19 February 2018 | Revised: 22 May 2018 | Accepted: 26 May 2018

DOI: 10.1002/wics.1443

WILEY WIREs COMPUTATIONAL STATISTICS

ADVANCED REVIEW

Spatial modeling with R-INLA: A review

Haakon Bakka[1] | Håvard Rue[1] | Geir-Arne Fuglstad[2] | Andrea Riebler[2] | David Bolin[3] | Janine Illian[4] | Elias Krainski[5] | Daniel Simpson[6] | Finn Lindgren[7]

[1]CEMSE Division, King Abdullah University of Science and Technology, Thuwal, Saudi Arabia

[2]Department of Mathematical Sciences, Norwegian University of Science and Technology, Trondheim, Norway

[3]Department of Mathematical Sciences, Chalmers University of Technology and University of Gothenburg, Gothenburg, Sweden

[4]CREEM, School of Mathematics and Statistics, University of St Andrews, St. Andrews, UK

[5]Departamento de Estatística, Universidade Federal do Paraná, Paraná, Brazil

[6]Department of Statistical Sciences, University of Toronto, Toronto, Canada

[7]School of Mathematics, University of Edinburgh, Edinburgh, UK

Correspondence
Håvard Rue, CEMSE Division, King Abdullah University of Science and Technology, Thuwal, Saudi Arabia.
Email: haavard.rue@kaust.edu.sa

Coming up with Bayesian models for spatial data is easy, but performing inference with them can be challenging. Writing fast inference code for a complex spatial model with realistically-sized datasets from scratch is time-consuming, and if changes are made to the model, there is little guarantee that the code performs well. The key advantages of R-INLA are the ease with which complex models can be created and modified, without the need to write complex code, and the speed at which inference can be done even for spatial problems with hundreds of thousands of observations. R-INLA handles latent Gaussian models, where fixed effects, structured and unstructured Gaussian random effects are combined linearly in a linear predictor, and the elements of the linear predictor are observed through one or more likelihoods. The structured random effects can be both standard areal model such as the Besag and the BYM models, and geostatistical models from a subset of the Matérn Gaussian random fields. In this review, we discuss the large success of spatial modeling with R-INLA and the types of spatial models that can be fitted, we give an overview of recent developments for areal models, and we give an overview of the stochastic partial differential equation (SPDE) approach and some of the ways it can be extended beyond the assumptions of isotropy and separability. In particular, we describe how slight changes to the SPDE approach leads to straight-forward approaches for nonstationary spatial models and nonseparable space–time models.

Thank you · شكرا

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