Project2

Gaussian random field with application of INLA

Yaolin Ge, Florian Beiser

Contents

Part I Multivariate normal distribution

Let $\boldsymbol{x} = (x_1, \ldots, x_n), n = 100$ be multivariate normal distributed with $E(x_i) = 0, Var(x_i) = 1$, and $Corr(x_i, x_j) = e^{-0.1|i-j|}$

a) Compute and image the covariance matrix $\boldsymbol{\Sigma}$ of $\boldsymbol{x}$

b) Find the lower Cholesky factor $\boldsymbol{L}$, such that $\boldsymbol{L}\boldsymbol{L}^T = \boldsymbol{\Sigma}$, of this covariance matrix, and image.

c) Sample $\boldsymbol{x} = \boldsymbol{L}\boldsymbol{z}$, where $\boldsymbol{z}$ is a length n random vector of independent standard normal variables. Plot the sample.

d) Find the precision matrix $\boldsymbol{Q}$ of the covariance matrix, and compute the lower Cholesky factor $\boldsymbol{L}_Q$, such that $\boldsymbol{L}_Q\boldsymbol{L}_Q^T = \boldsymbol{Q}$, of this matrix. Image these matrices and compare them to the images obtained in a) and b)

e) Sample $\boldsymbol{x}$ by solving $\boldsymbol{L}_Q^T\boldsymbol{x} = \boldsymbol{z}$, where $\boldsymbol{z}$ is a length n random vector of independen standard normal variables. Plot the sample.

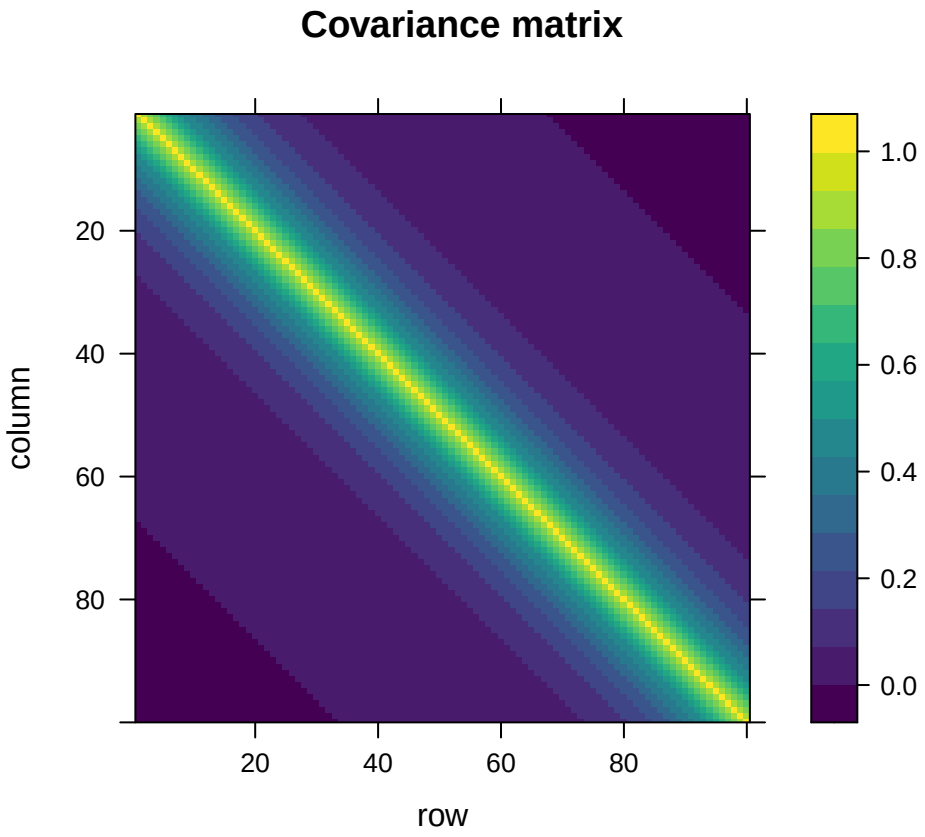
f) Permute the ordering of variables in $\boldsymbol{x}$, and redo the exercises.

Solution to Part I

a)

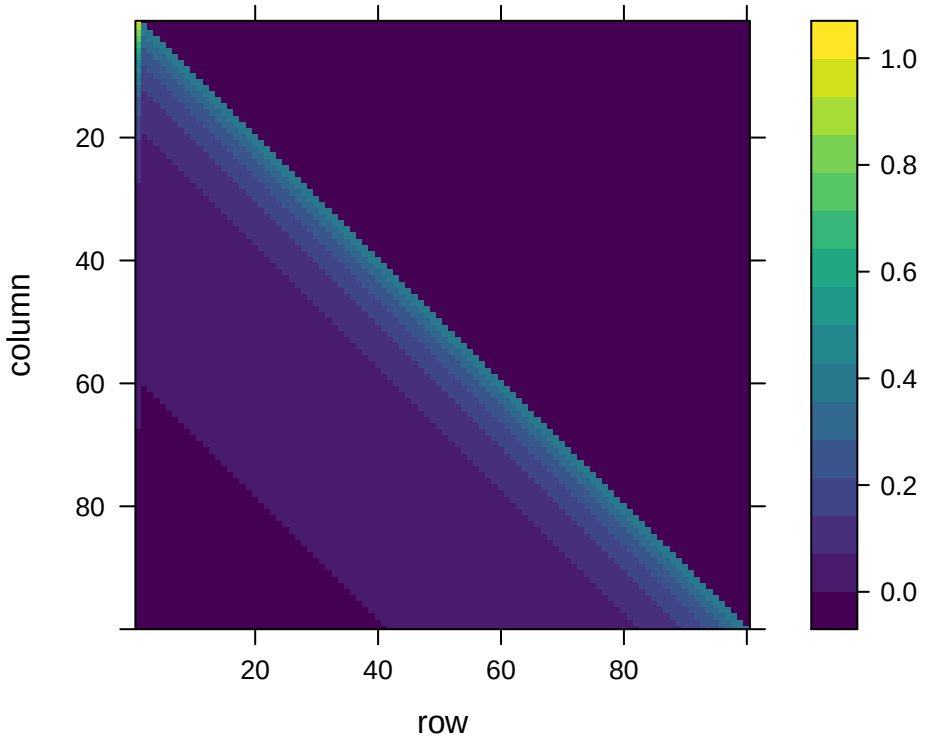
Given that $\mathbf{\Sigma} = e^{-0.1|i-j|}$. The covariance matrix can be expressed as follows:

$$\Sigma = \begin{pmatrix} 1 & e^{-0.1h_{12}} & \dots & e^{-0.1h_{1n}} \\ e^{-0.1h_{21}} & 1 & \dots & e^{-0.1h_{2n}} \\ \vdots & \vdots & \ddots & \vdots \\ e^{-0.1h_{n1}} & e^{-0.1h_{n2}} & \dots & 1 \end{pmatrix}$$



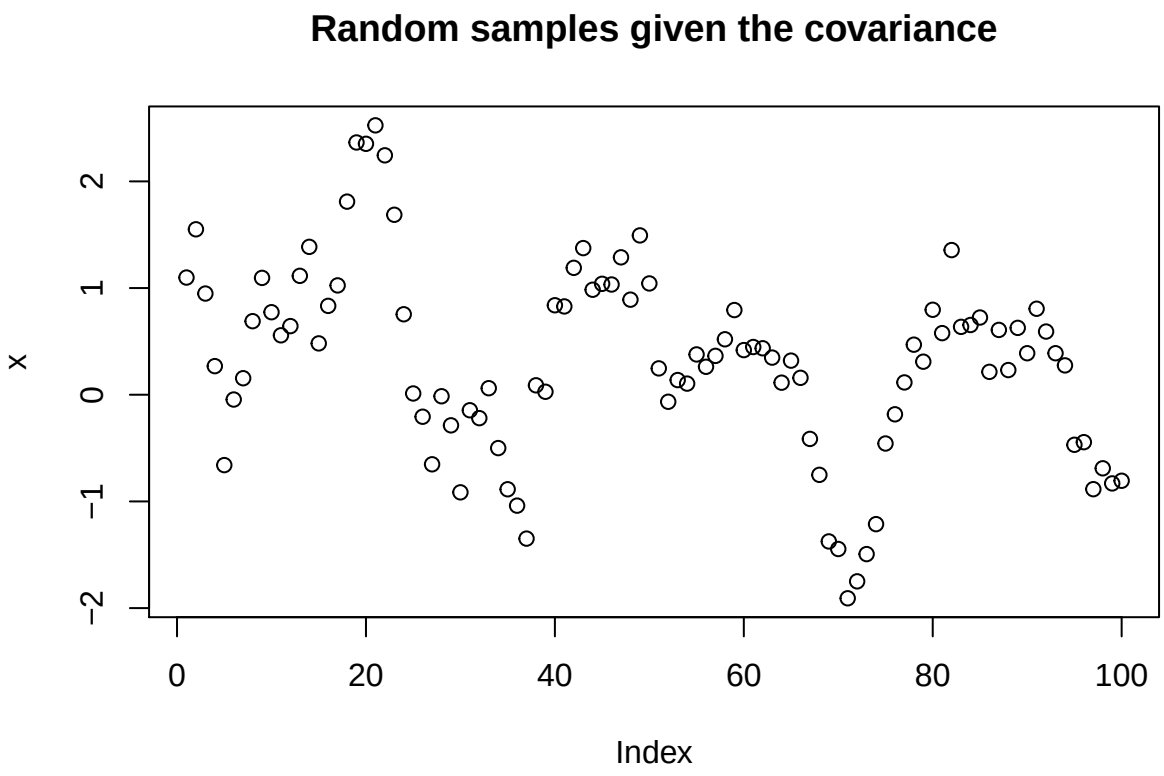
b)

According to the cholesky decomposition rule, $\boldsymbol{L}$ is the lower triangular matrix for $\mathbf{\Sigma}$, it can be easily computed from R using `L = chol(Sigma)`. It is then plotted as below.



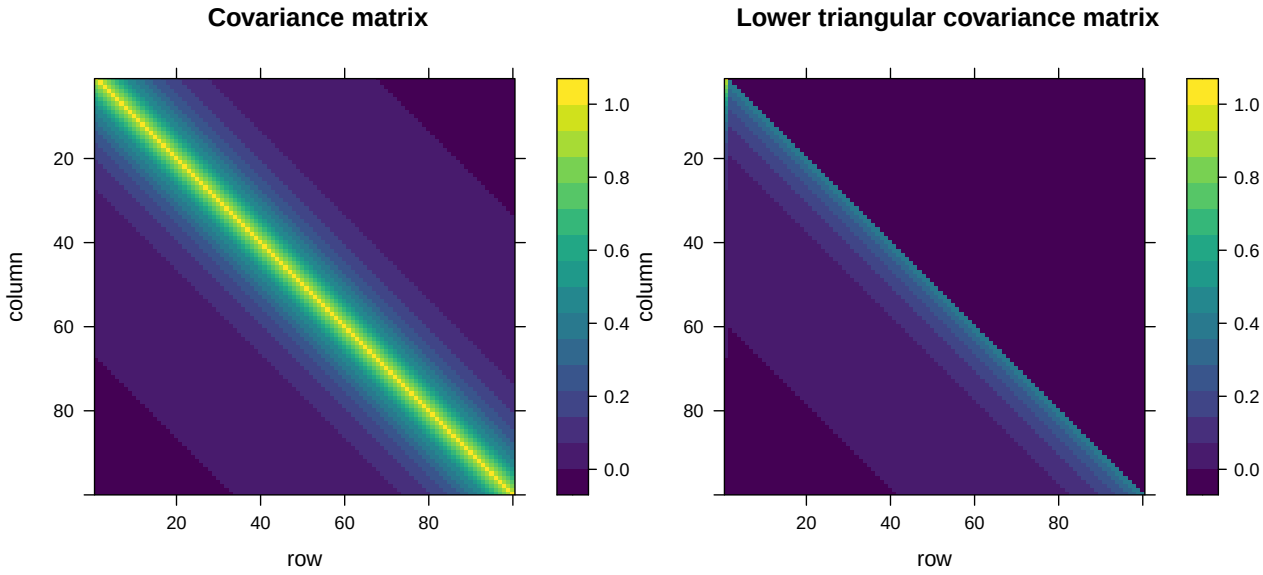
c)

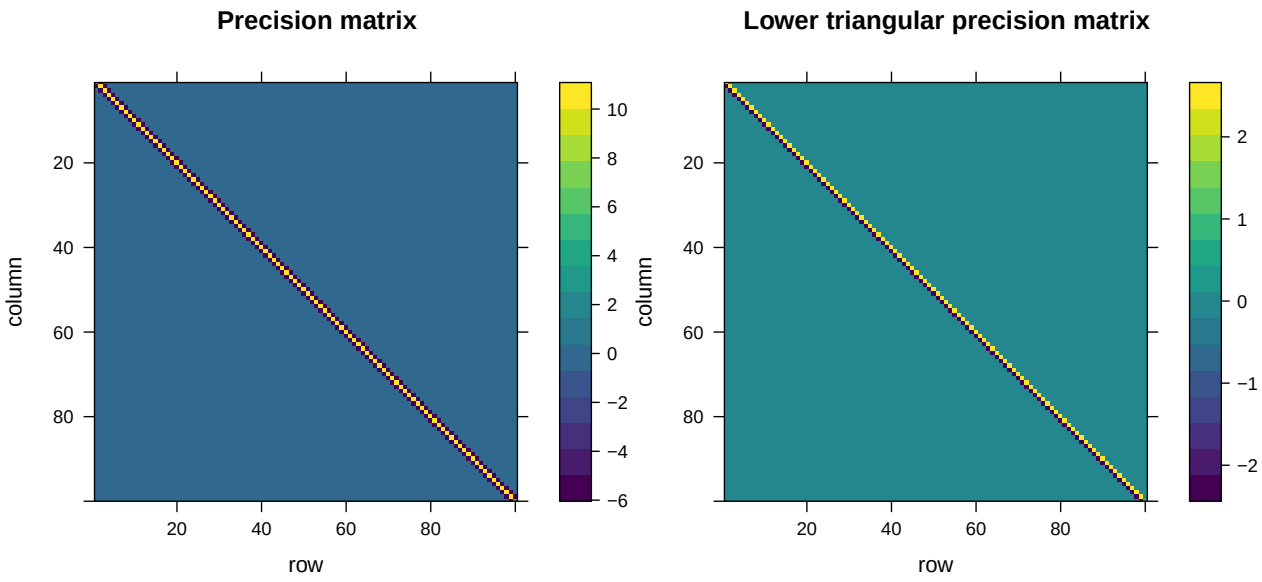
Sample using $\boldsymbol{x} = \boldsymbol{L}\boldsymbol{z}$ transforms the zero-mean, standard normal random variales to the random variables with the desired covariance matrix.



d)

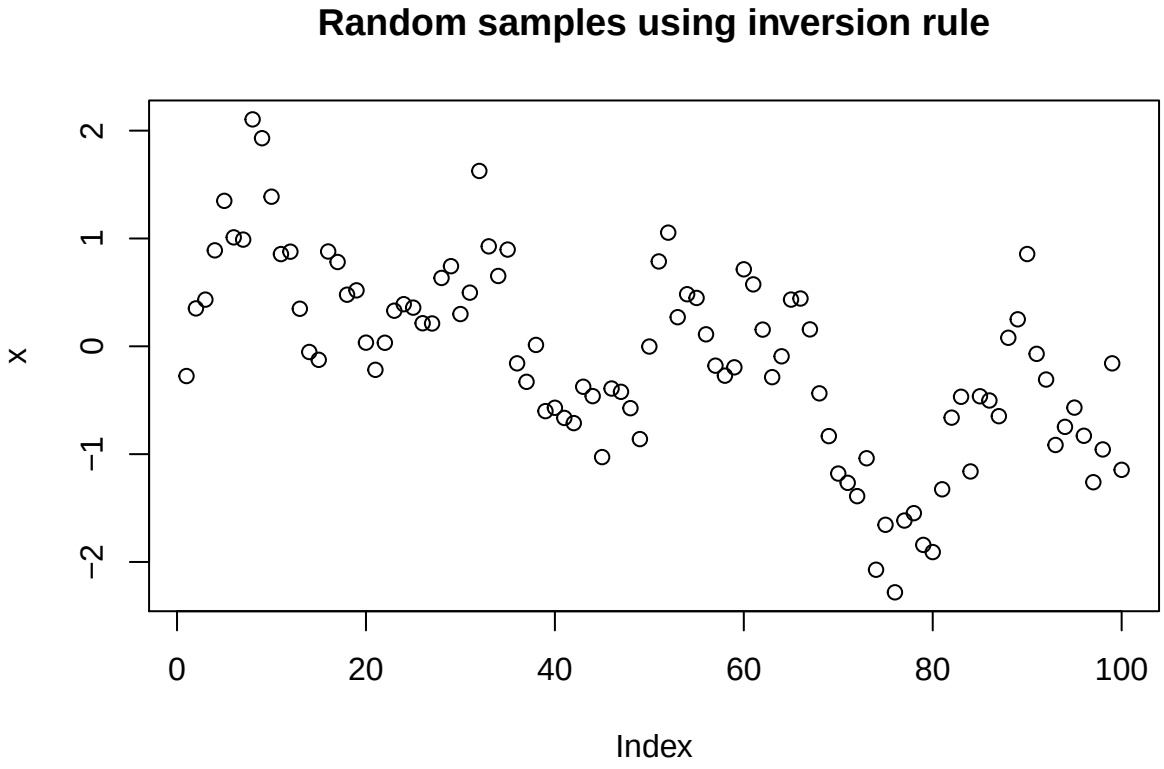
The precision matrix $\boldsymbol{Q}$ is the inverse of the covariance matrix $\boldsymbol{\Sigma}$, it is computed using `Q = solve(Sigma)` in R. The three matrices are thereby depicted as follows. Since the covariance matrix is not singular, given that the it belongs to the Matern family, thus it is analytically guaranteed to have positive definite property. Therefore, both precision matrix and the lower triangular precision matrix exist.





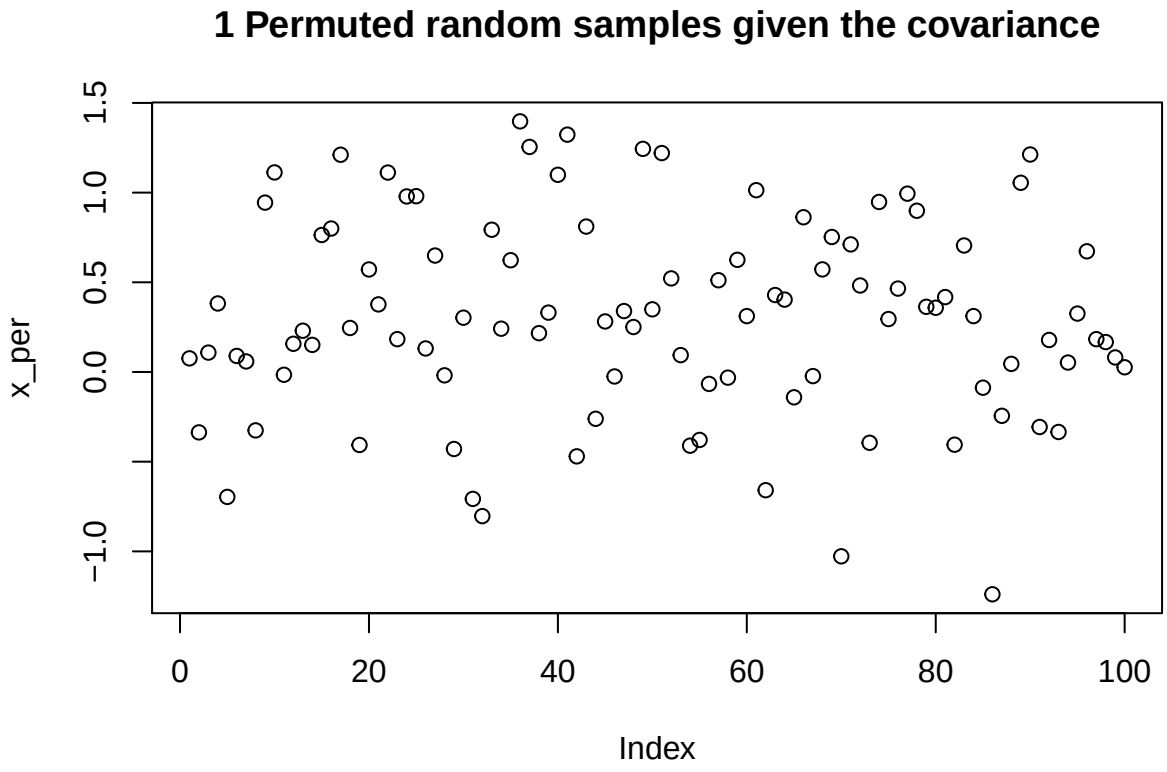
e)

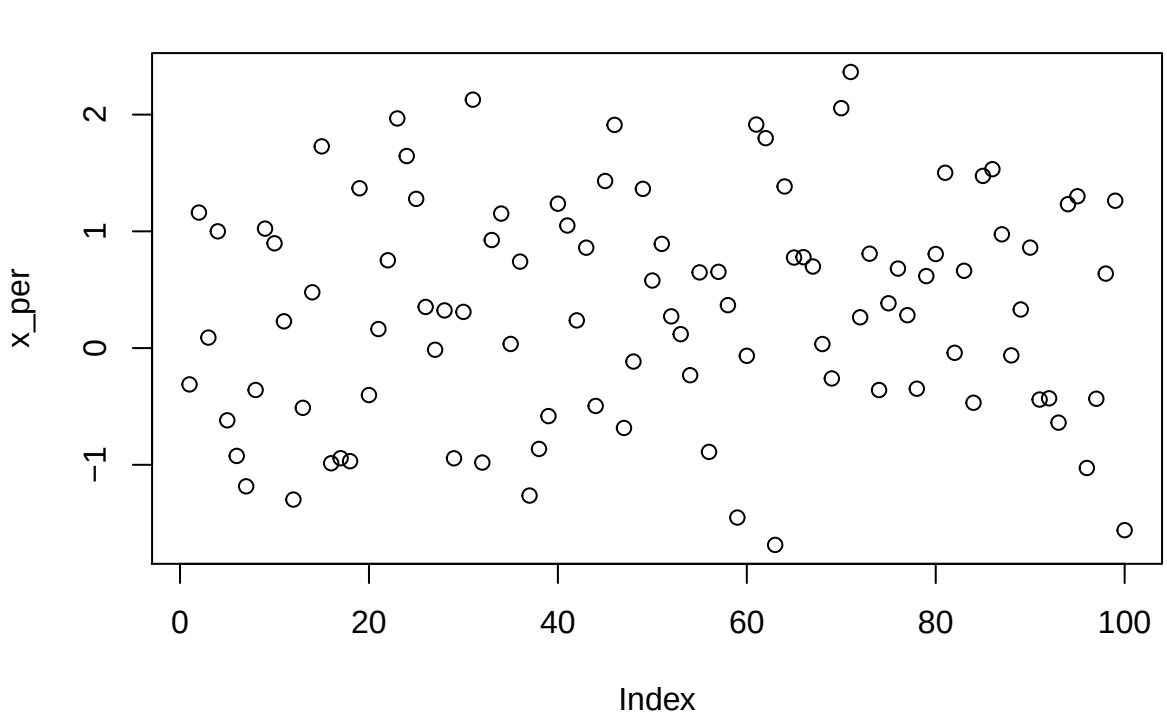
Similarly, the expected random samples can be generated using the inversion of the above formula, thus $\boldsymbol{L}_Q^T \boldsymbol{x} = \boldsymbol{z}$

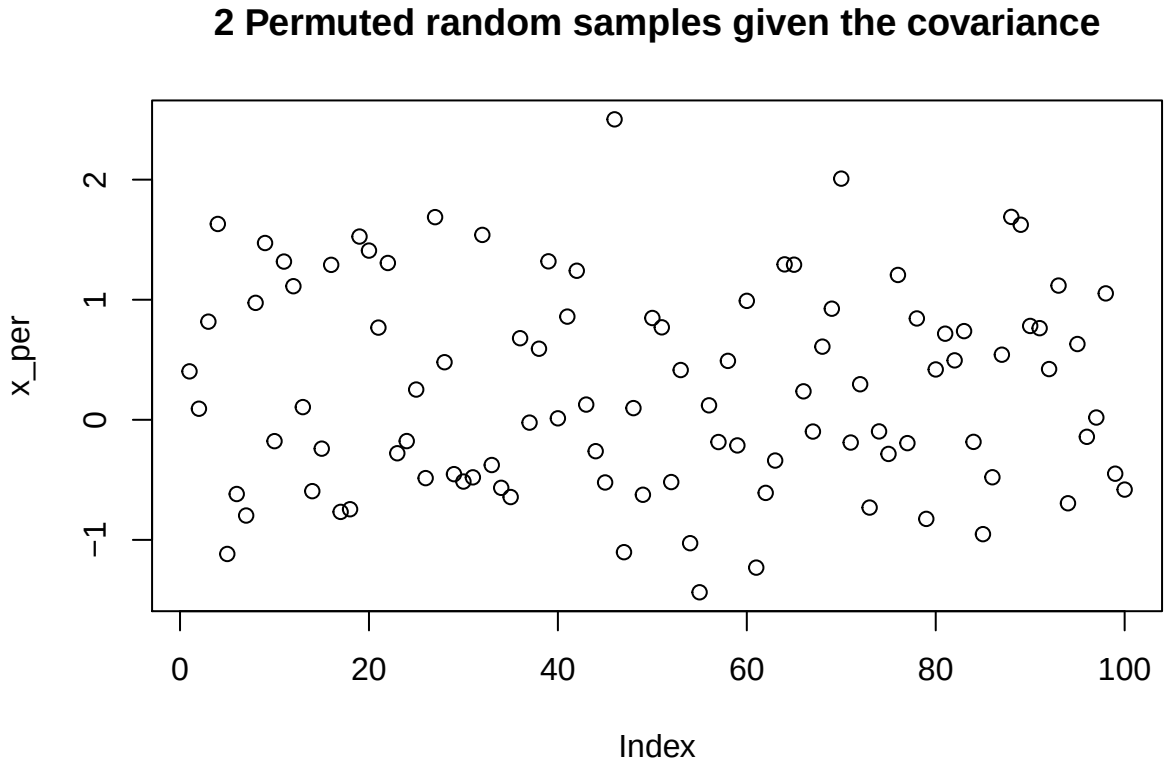


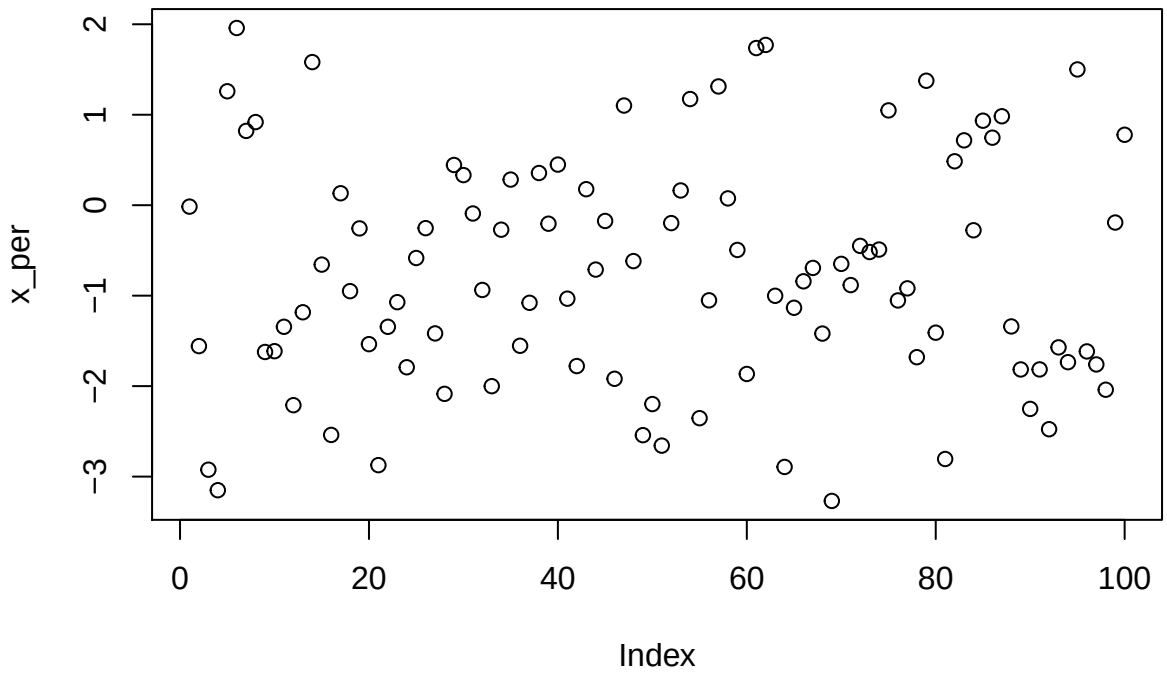
f)

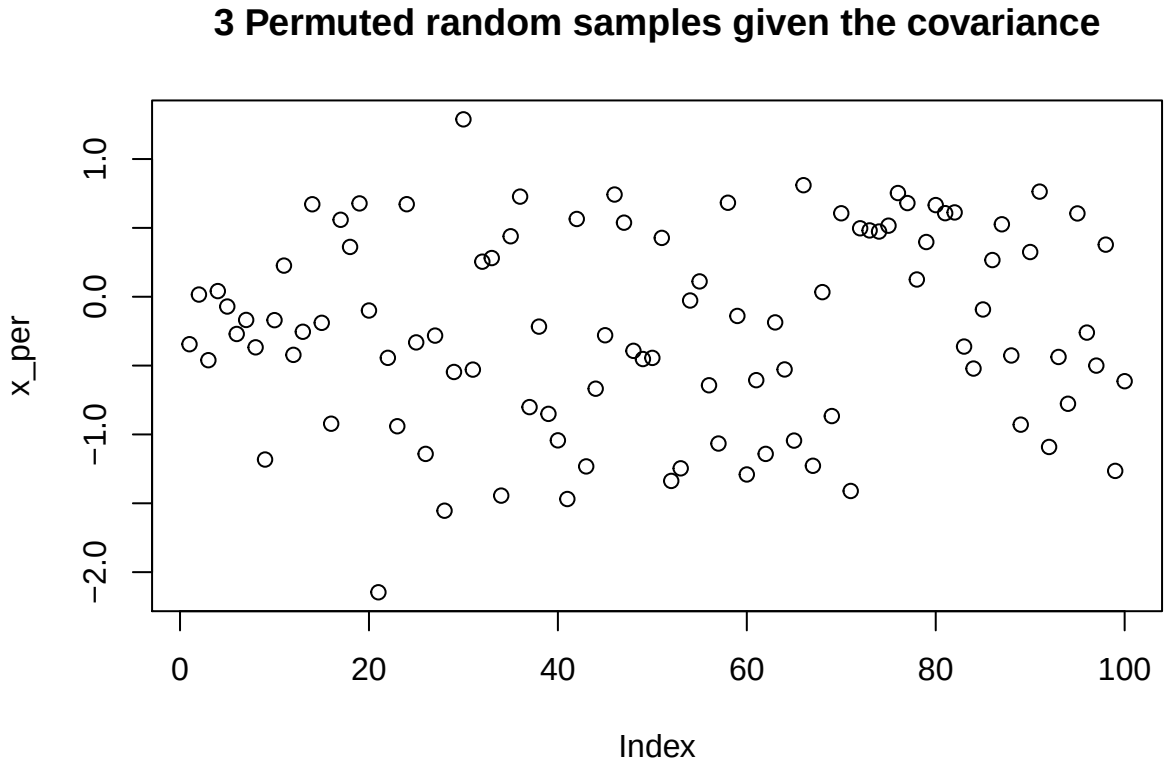
Permute $\boldsymbol{x}$ to make randomise the ordering of the grid, the asscociated covariance matrix can be thereby modified in a sparse way.

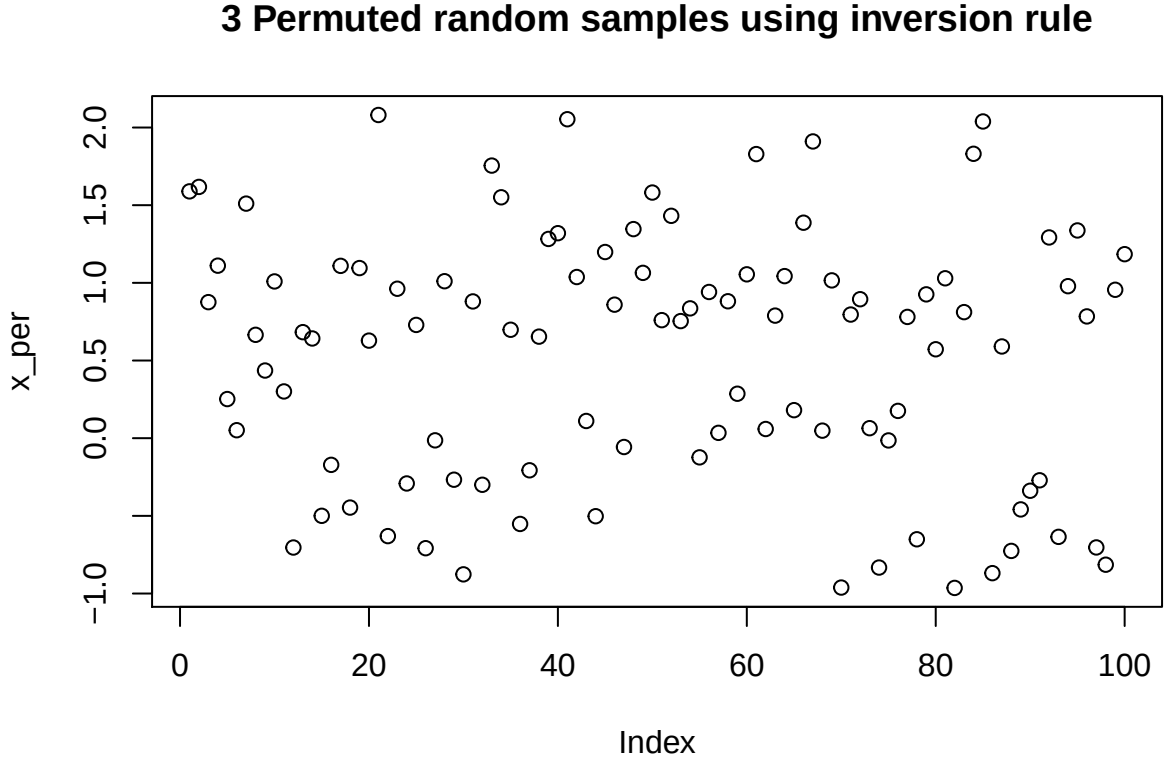

1 Permuted random samples given the covariance
x_per
1.5
1.0
0.5
0.0
−1.0
0
20
40
60
80
100
Index

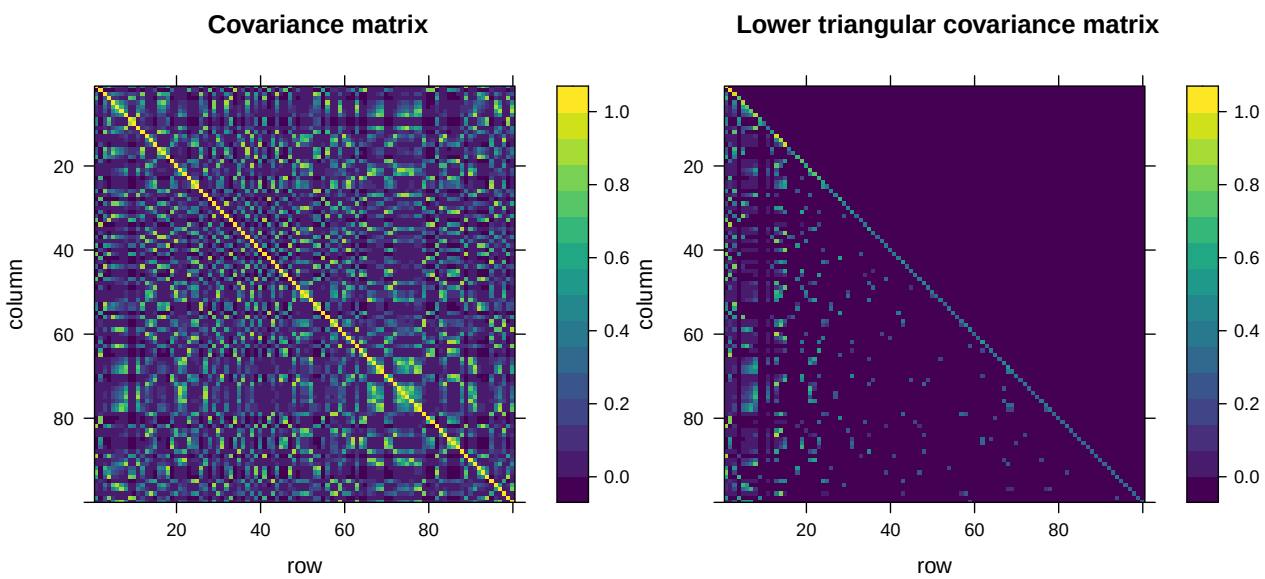

1 Permuted random samples using inversion rule
x_per
2
1
0
−1
0
20
40
60
80
100
Index

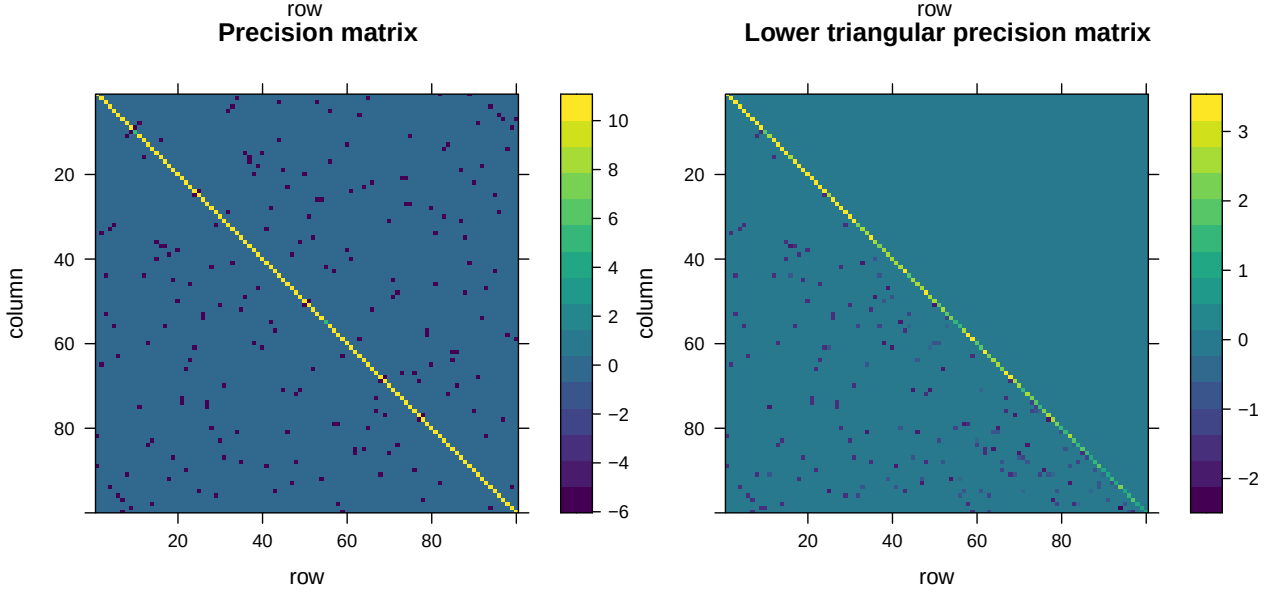

2 Permuted random samples given the covariance
x_per
Index
2
1
0
−1
0
20
40
60
80
100

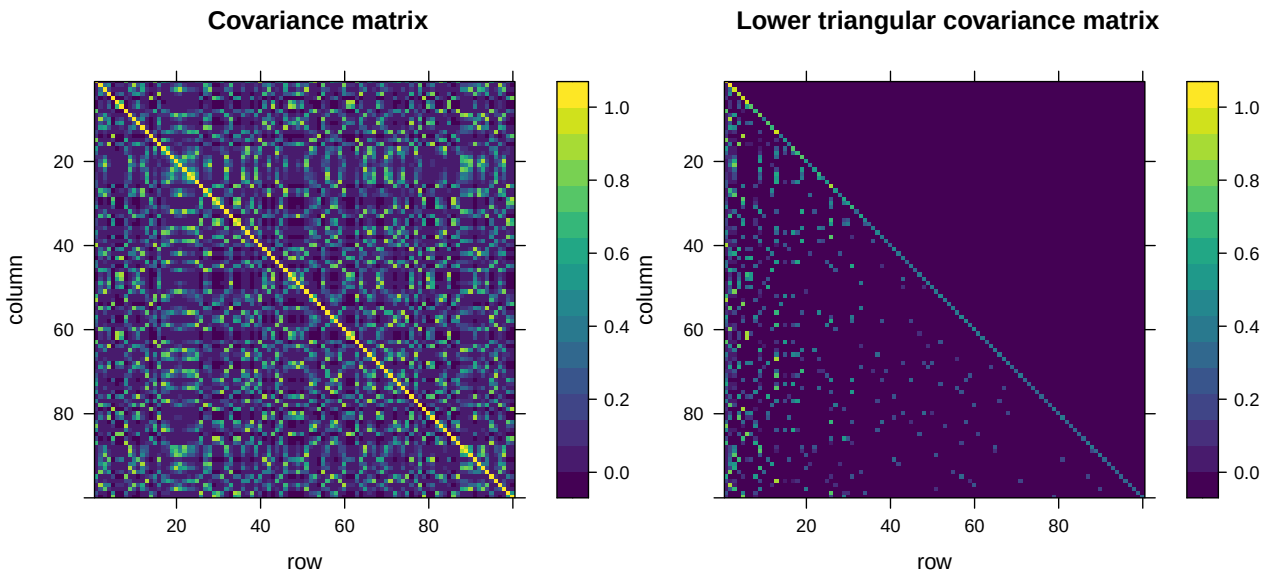

2 Permuted random samples using inversion rule
x_per
Index
2
1
0
−1
−2
−3
0
20
40
60
80
100

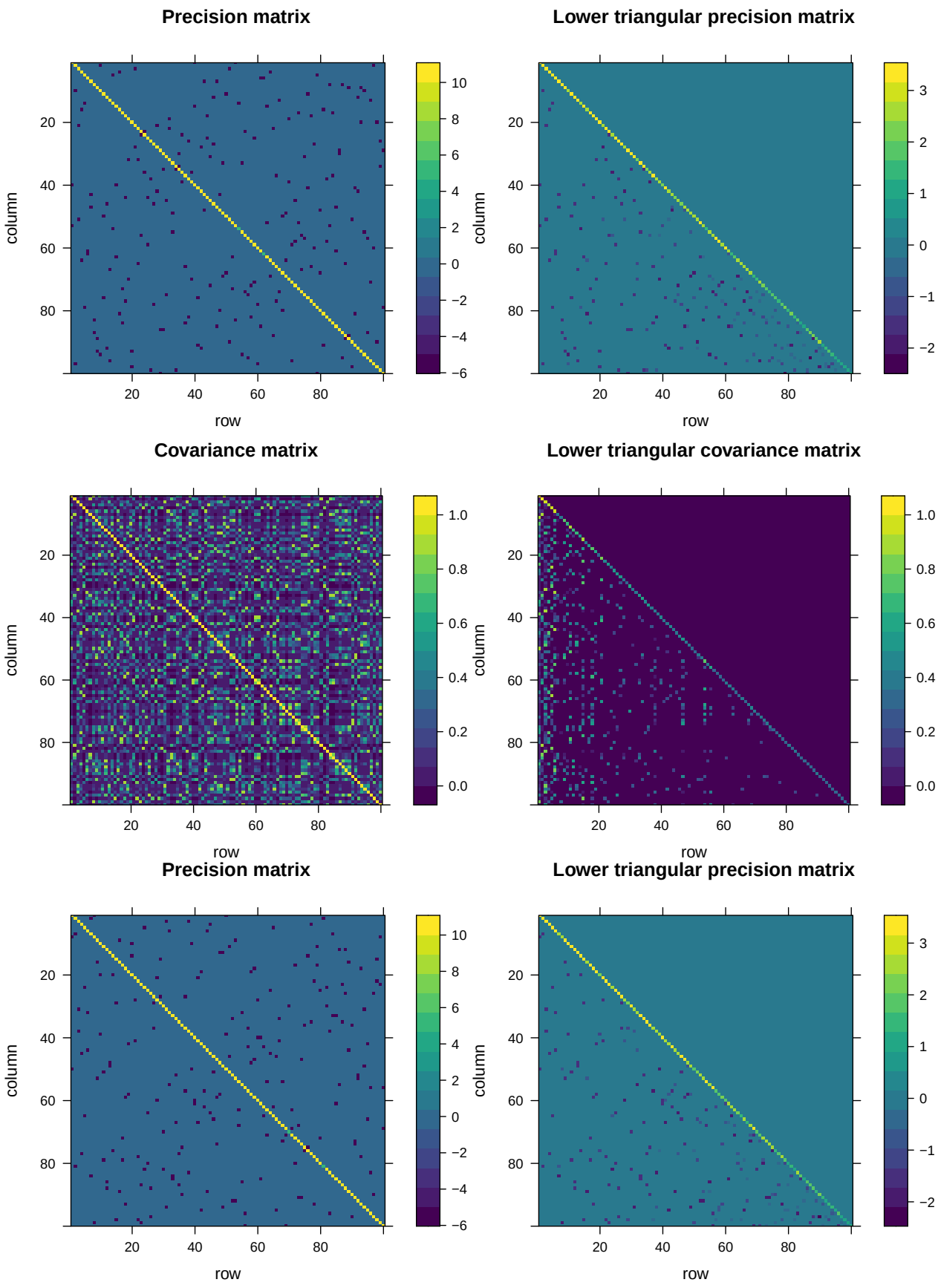

3 Permuted random samples given the covariance
x_per
1.0
0.0
−1.0
−2.0
0
20
40
60
80
100
Index


3 Permuted random samples using inversion rule
x_per
2.0
1.5
1.0
0.5
0.0
−1.0
0
20
40
60
80
100
Index


Covariance matrix
Lower triangular covariance matrix
column
row
20
40
60
80
1.0
0.8
0.6
0.4
0.2
0.0


Precision matrix
Lower triangular precision matrix
column
row
20
40
60
80
10
8
6
4
2
0
−2
−4
−6
3
2
1
0
−1
−2


Covariance matrix
Lower triangular covariance matrix
column
row
20
40
60
80
1.0
0.8
0.6
0.4
0.2
0.0


Precision matrix
Lower triangular precision matrix
Covariance matrix
Lower triangular covariance matrix
Precision matrix
Lower triangular precision matrix
column
row
20
40
60
80
10
8
6
4
2
0
−2
−4
−6
3
1
−1
1.0
0.8
0.6
0.4
0.2
0.0

Part II Gaussian random fields and Kriging

The purpose of this computer exercise is to give an introduction to parameter estimation and kriging for Gaussian random field models for spatial data.

We assume the following observation model on the unit square:

$$y(\boldsymbol{s}_j) = x(\boldsymbol{s}_j) + \epsilon_j, \quad j = 1, \ldots, N,$$

where $\epsilon_j \;\; N(0, \tau^2)$ are independent measurement noise terms. Further, consider a Matérn covariance function for the Gaussian random field $\boldsymbol{x(s)}$:

$$Cov(x(\boldsymbol{s}_i), x(\boldsymbol{s}_j)) = \Sigma_{i,j} = \sigma^2(1 + \phi h)\exp(-\phi h),$$

where h denotes the Euclidean distance between the two sites $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$.

We assume the mean increases with east and north coordinates as follows: $\mu_j = \alpha((s_{j1} - 0.5) + (s_{j2} - 0.5))$, for site $\boldsymbol{s}_j = (s_{j1}, s_{j2})$ on the unit square.

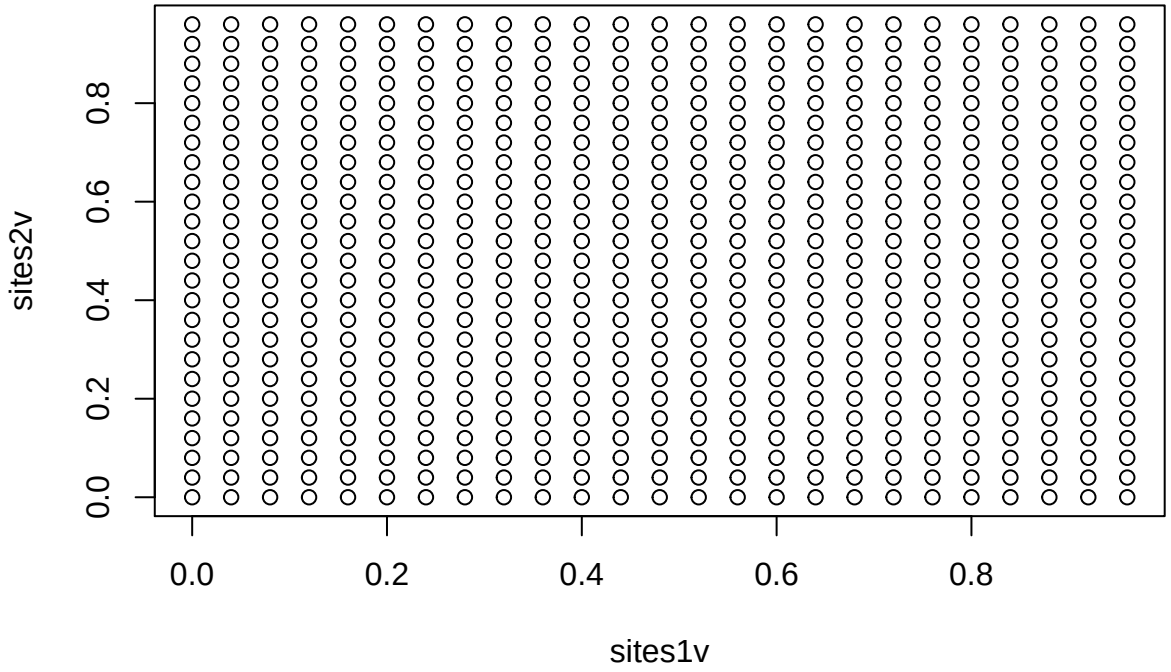
2.1 Simulation

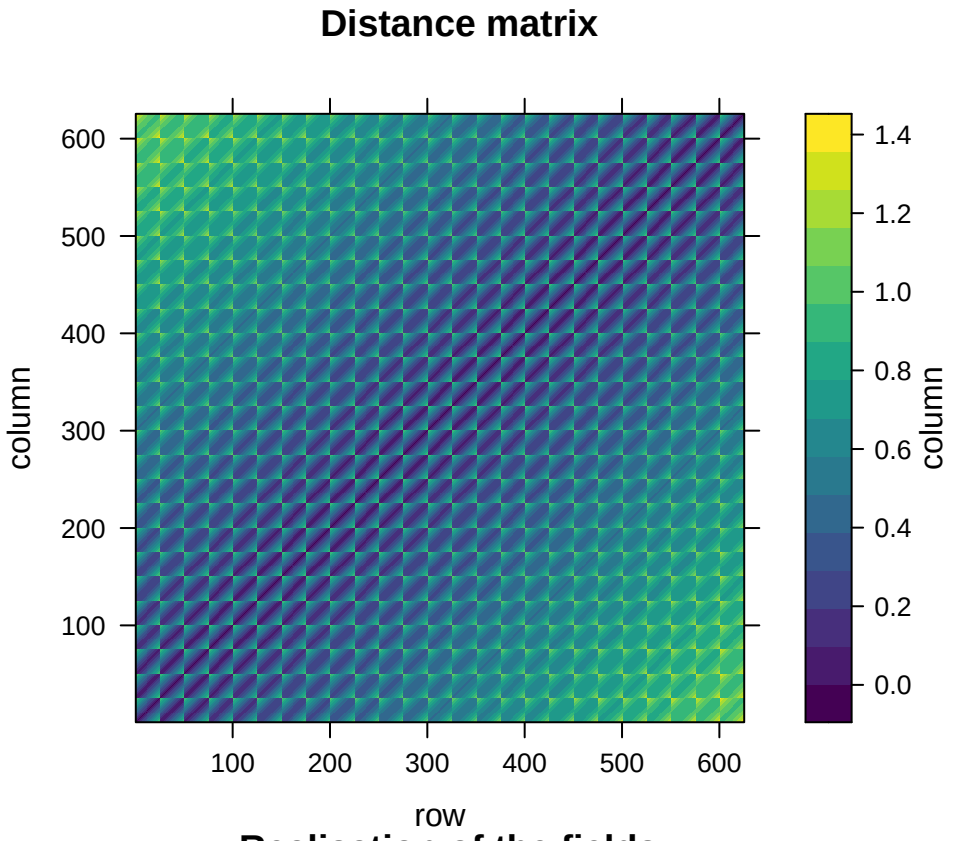
Simulate $N = 200$ random sites in the unit square and plot them. Form the covariance matrix using $\sigma = 1, \phi = 10, \tau = 0.05$. Take its Cholesky decomposition and simulate dependent zero-mean Gaussian data variables, then add the mean using $\alpha = 1$. Plot your observations.

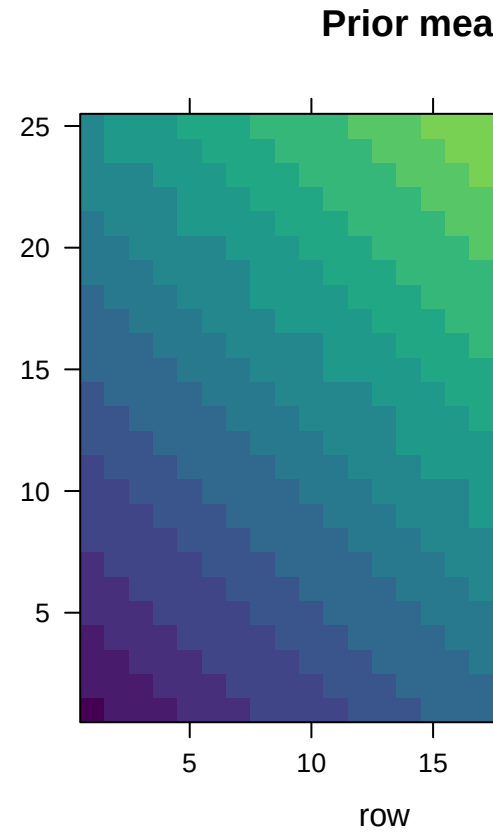
The true mean of the field is expressed as

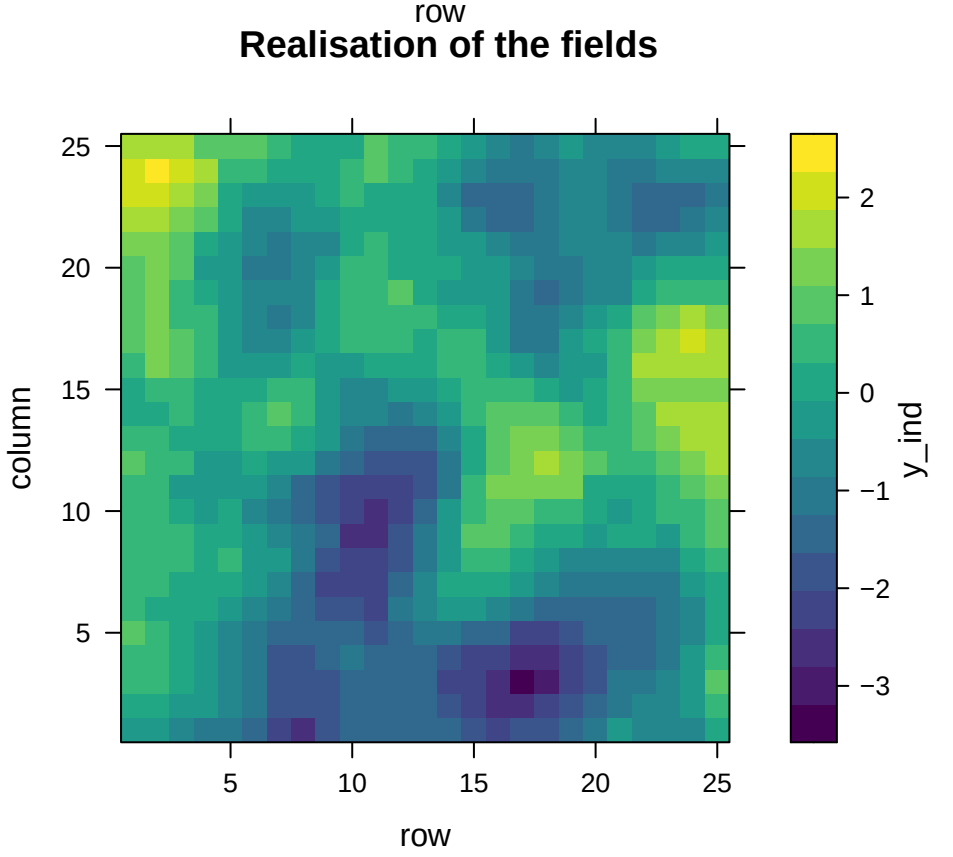
$$\mu_i = \alpha((s_{i1} - 0.5) + (s_{i2} - 0.5)) = -\alpha + \alpha s_{i1} + \alpha s_{i2} = \underbrace{\begin{bmatrix} 1 & s_{i1} & s_{i2} \end{bmatrix}}_{\boldsymbol{h}^T(\boldsymbol{s}_i)} \underbrace{\begin{bmatrix} -\alpha \\ \alpha \\ \alpha \end{bmatrix}}_{\boldsymbol{\beta}}$$

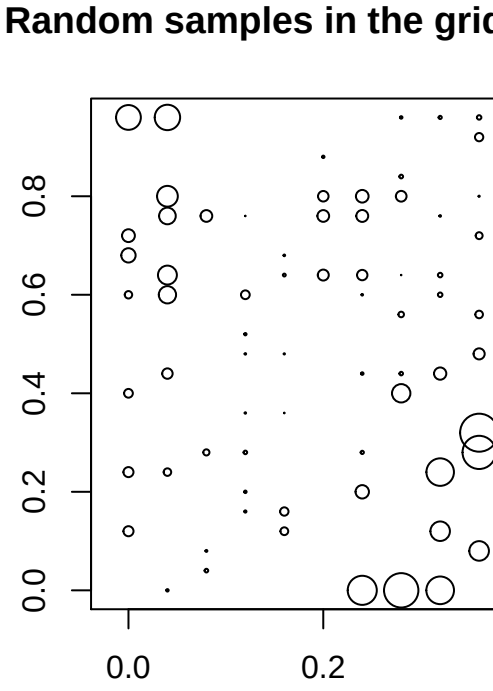
where s_{i1}, s_{i2} are the location from east and north direction in the grid.




Distance matrix
column
row
600
500
400
300
200
100
100
200
300
400
500
600
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0


Prior mea
column
row
25
20
15
10
5
5
10
15


Realisation of the fields
column
row
25
20
15
10
5
5
10
15
20
25
2
1
0
−1
−2
−3
y_ind


Random samples in the gric
0.0
0.2
0.4
0.6
0.8
0.0
0.2

2.2 Paramter estimation

We will now use the simulated data to estimate the model parameters $\alpha, \sigma^2, \tau^2, \phi$ using maximum likelihood estimation. Iterate between the update for the mean parameter, and updating the covariance parameters. Monitor the likelihood function at each step of the algorithm to check convergence. Since the sampling randomly in the field can cause problems in the distance matrix and accordingly the rest of the calculations. Therefore, it is more stable to use the random samples from the regular grids.

The mean of the field is modelled by $p(\boldsymbol{x})$ and the imperfect information $\boldsymbol{y} = (y_1, \ldots, y_m)$ conditional on $\boldsymbol{x}$ can be modelled by $p(\boldsymbol{y}|\boldsymbol{x})$, which can be expressed as follows:

$$p(\boldsymbol{x}) = N(\boldsymbol{H}\beta, \boldsymbol{\Sigma}), \quad p(\boldsymbol{y}|\boldsymbol{x}) = N(\boldsymbol{F}\boldsymbol{x}, \boldsymbol{T})$$

Therefore, the marginal likelihood of the data is

$$p(\boldsymbol{y}) = N(\boldsymbol{G}\boldsymbol{\beta}, \boldsymbol{C}), \quad \boldsymbol{G} = \boldsymbol{F}\boldsymbol{H}, \quad \boldsymbol{C} = \boldsymbol{F}\boldsymbol{\Sigma}\boldsymbol{F}^{\boldsymbol{T}} + \boldsymbol{T}$$

The log-likelihood as a function of β and unknown fixed nuisance parameters $\boldsymbol{\theta}$ in the prior covariance matrix $\boldsymbol{\Sigma} = \boldsymbol{\Sigma}(\boldsymbol{\theta})$, and/or the likelihood noise matrix $\boldsymbol{T} = \boldsymbol{T}(\boldsymbol{\theta})$ becomes

$$l(\boldsymbol{\theta}, \boldsymbol{\beta}) = -\frac{m}{2}\log(2\pi) - \frac{1}{2}\log|\boldsymbol{C}| - \frac{1}{2}(\boldsymbol{y} - \boldsymbol{G}\boldsymbol{\beta})^T\boldsymbol{C}^{-1}(\boldsymbol{y} - \boldsymbol{G}\boldsymbol{\beta})$$

The MLEs of $\boldsymbol{\beta}$ and $\boldsymbol{\theta}$ are obtained by

$$(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\theta}}) = \arg\max_{\boldsymbol{\beta}, \boldsymbol{\theta}}\{l(\boldsymbol{\beta}, \boldsymbol{\theta})\}$$

For fixed $\boldsymbol{\theta}$, the MLEs of $\boldsymbol{\beta}$ can be determined analytically.

$$\frac{dl}{d\boldsymbol{\beta}} = \boldsymbol{G}^T\boldsymbol{C}^{-1}\boldsymbol{y} - \boldsymbol{G}^T\boldsymbol{C}^{-1}\boldsymbol{G}\boldsymbol{\beta} = \boldsymbol{0}, \quad \hat{\boldsymbol{\beta}} = (\boldsymbol{G}^T\boldsymbol{C}^{-1}\boldsymbol{G})^{-1}\boldsymbol{G}^T\boldsymbol{C}^{-1}\boldsymbol{y}$$

Whereas for fixed $\boldsymbol{\beta}$, the MLE of nuisance parameters $\boldsymbol{\theta}$ can be obtained by numerical maximization. Let $\boldsymbol{z} = \boldsymbol{y} - \boldsymbol{G}\boldsymbol{\beta}$, and $\boldsymbol{Q} = \boldsymbol{C}^{-1}$. For each component of $\boldsymbol{\theta}_r$, $r = 1, \ldots, d$, in this case, θ has 3 components (σ^2, η, τ^2). The score of the log-likelihood becomes

$$\frac{dl}{d\theta_r} = -\frac{1}{2}\text{trace}(\boldsymbol{Q}\frac{d\boldsymbol{C}}{d\theta_r}) + \frac{1}{2}\boldsymbol{z}^T\boldsymbol{Q}\frac{d\boldsymbol{C}}{d\theta_r}\boldsymbol{Q}\boldsymbol{z}$$

The above mentioned score can be solved iteratively using Fisher scoring algorithm. To achieve the numerical stability, the expected Hessian is applied, which is

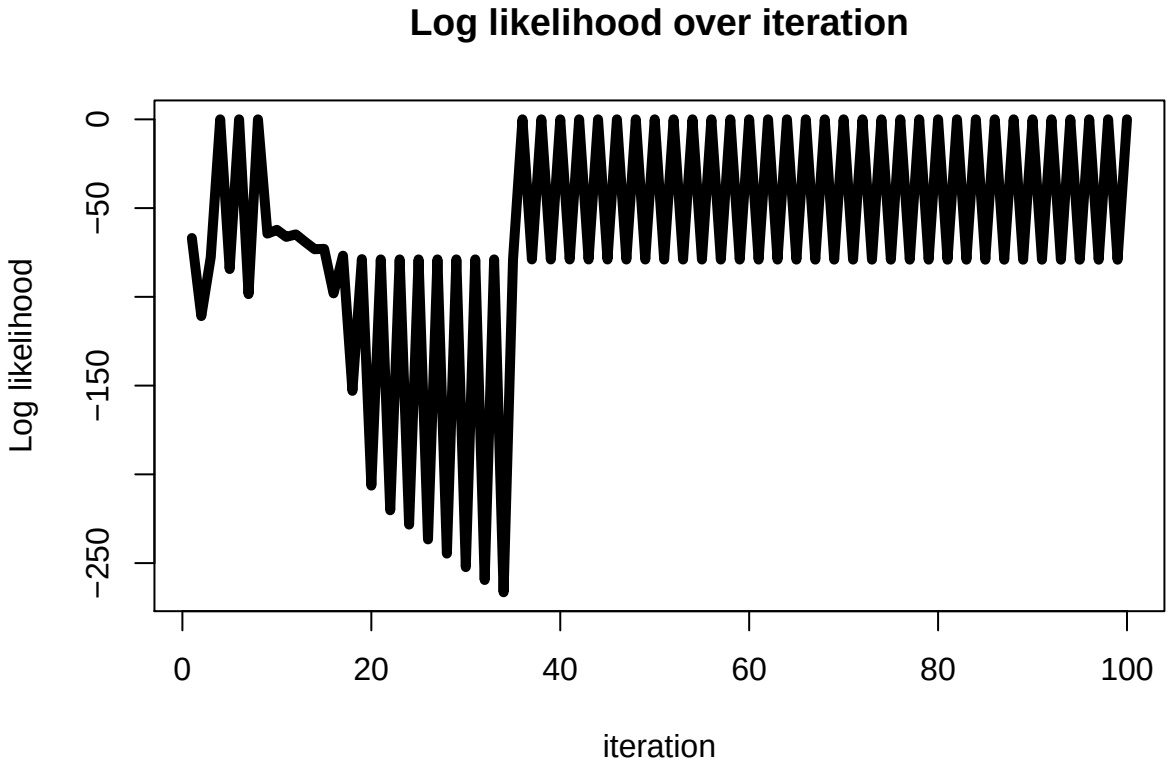
$$E(\frac{d^2l}{d\theta_r d\theta_{\bar{r}}}) = -\frac{1}{2}\text{trace}(\boldsymbol{Q}\frac{d\boldsymbol{C}}{d\theta_{\bar{r}}}\boldsymbol{Q}\frac{d\boldsymbol{C}}{d\theta_r})$$

The pseudo code for the Fisher scoring algorithm can then be expressed as follows:

Data: initial β_0, θ_0
Result: Converged $\hat{\beta}$, $\hat{\theta}$
while *not converged* **do**
  $\boldsymbol{C} = \boldsymbol{C}(\boldsymbol{\theta}^b)$;
  $\boldsymbol{\beta}^{b+1} = [\boldsymbol{G}^T\boldsymbol{C}^{-1}\boldsymbol{G}]^{-1}\boldsymbol{G}^T\boldsymbol{C}^{-1}\boldsymbol{y}$;
  $\boldsymbol{z} = \boldsymbol{y} - \boldsymbol{G}\boldsymbol{\beta}^{b+1}$;
  $\boldsymbol{Q} = \boldsymbol{C}^{-1}$;
  $\boldsymbol{C}_r^* = \frac{d\boldsymbol{C}(\boldsymbol{\theta}^b)}{d\theta_r}, \quad r = 1, \ldots, d$;
  $u_r = \frac{dl}{d\theta_r} = -\frac{1}{2}\text{trace}(\boldsymbol{Q}\boldsymbol{C}_r^*) + \frac{1}{2}\boldsymbol{z}^T\boldsymbol{Q}\boldsymbol{C}_r^*\boldsymbol{Q}\boldsymbol{z}$;
  $\boldsymbol{V}_{r\bar{r}} = E(\frac{d^2l}{d\theta_r d\theta_{\bar{r}}}) = -\frac{1}{2}\text{trace}(\boldsymbol{Q}\boldsymbol{C}_r^*\boldsymbol{Q}\boldsymbol{C}_{\bar{r}}^*)$;
  $\boldsymbol{\theta}^{b+1} = \boldsymbol{\theta}^b + \boldsymbol{V}^{-1}\boldsymbol{u}$;
  $b = b + 1$
end

```
## [1] "0.0498514242905616  , iter no is  0"
## [1] "0.0791980627080858  , iter no is  1"
## [1] "0.107779649936967  , iter no is  2"
## [1] "0.168730096868739  , iter no is  3"
## [1] "0.181488954087671  , iter no is  4"
## [1] "0.183627764711045  , iter no is  5"
## [1] "0.315935940656133  , iter no is  6"
## [1] "0.112306426919098  , iter no is  7"
## [1] "0.159104200877777  , iter no is  8"
## [1] "0.0212197616928102  , iter no is  9"
## [1] "0.0260798823435196  , iter no is  10"
## [1] "0.0324756640309749  , iter no is  11"
## [1] "0.0391517453562815  , iter no is  12"
## [1] "0.0487265781543242  , iter no is  13"
## [1] "0.0569654917399516  , iter no is  14"
## [1] "0.0696725186840789  , iter no is  15"
## [1] "0.0776579965839025  , iter no is  16"
## [1] "0.0902434052952608  , iter no is  17"
## [1] "0.0950846789673355  , iter no is  18"
## [1] "0.101312503068609  , iter no is  19"
## [1] "0.103343885160729  , iter no is  20"
## [1] "0.103555283826031  , iter no is  21"
## [1] "0.105259808056863  , iter no is  22"
## [1] "0.104730724942536  , iter no is  23"
## [1] "0.106430977041045  , iter no is  24"
## [1] "0.105854408862935  , iter no is  25"
## [1] "0.107527425282265  , iter no is  26"
## [1] "0.106852371839486  , iter no is  27"
## [1] "0.108506773194866  , iter no is  28"
## [1] "0.10775174011884  , iter no is  29"
## [1] "0.109391175749325  , iter no is  30"
## [1] "0.108564627889718  , iter no is  31"
## [1] "0.110192574326133  , iter no is  32"
## [1] "0.109303082688636  , iter no is  33"
## [1] "0.110922230006541  , iter no is  34"
## [1] "0.109976955681716  , iter no is  35"
## [1] "0.111589439784168  , iter no is  36"
## [1] "0.110594515418375  , iter no is  37"
## [1] "0.112202001559578  , iter no is  38"
## [1] "0.111162627132339  , iter no is  39"
## [1] "0.112766409762504  , iter no is  40"
## [1] "0.111687010861518  , iter no is  41"
## [1] "0.113288093121402  , iter no is  42"
## [1] "0.112172453708435  , iter no is  43"
## [1] "0.113771613305496  , iter no is  44"
## [1] "0.112622992290343  , iter no is  45"
## [1] "0.114220833432155  , iter no is  46"
## [1] "0.113042060488477  , iter no is  47"
## [1] "0.114639054237379  , iter no is  48"
## [1] "0.113432607239147  , iter no is  49"
## [1] "0.115029122350268  , iter no is  50"
## [1] "0.113797188887614  , iter no is  51"
## [1] "0.115393515173824  , iter no is  52"
## [1] "0.114138041113925  , iter no is  53"
```

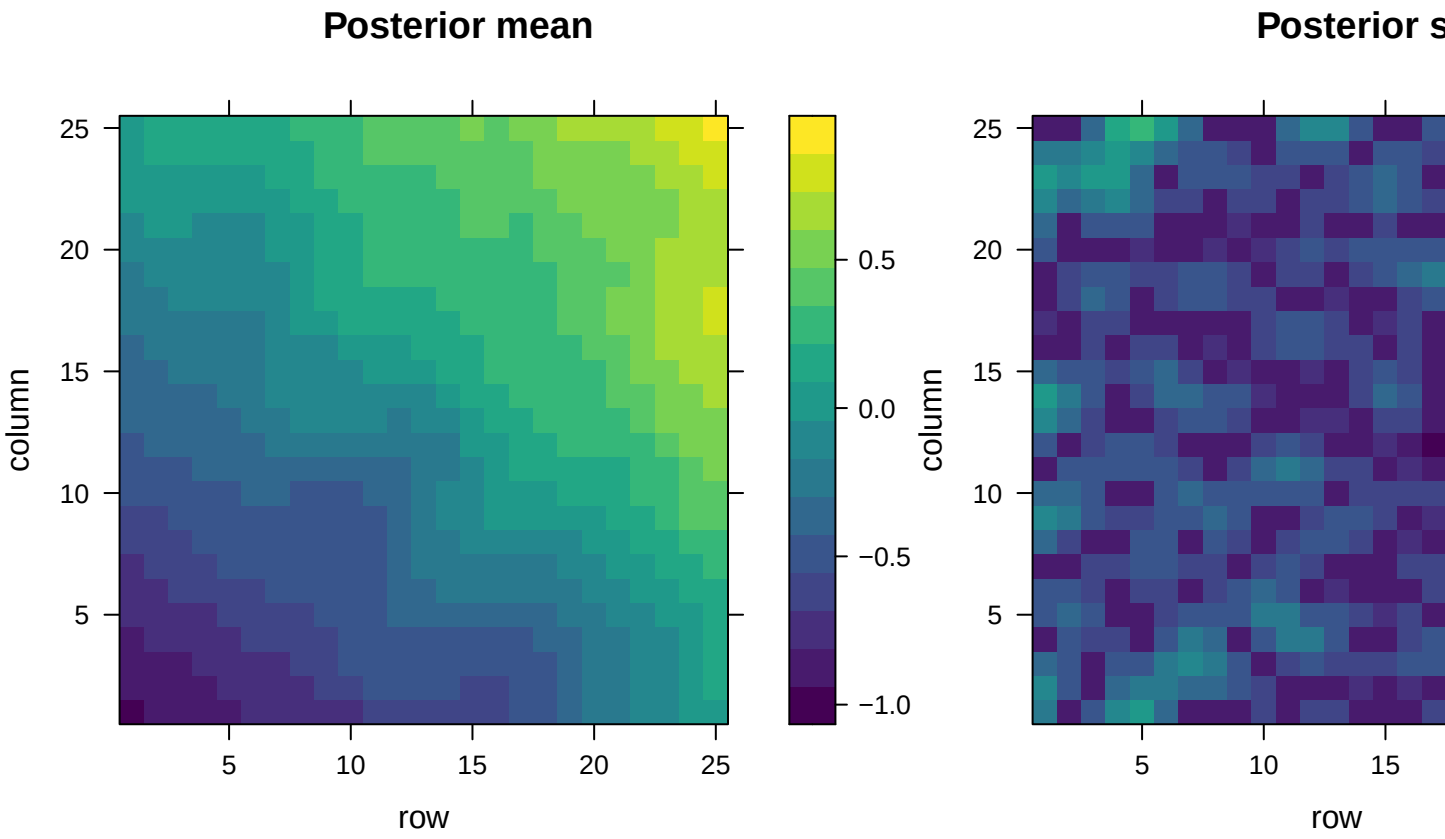
```
## [1] "0.11573440706655  , iter no is  54"
## [1] "0.114457134847082  , iter no is  55"
## [1] "0.116053720925596  , iter no is  56"
## [1] "0.114756219885832  , iter no is  57"
## [1] "0.116353168579147  , iter no is  58"
## [1] "0.115036859184427  , iter no is  59"
## [1] "0.116634282682392  , iter no is  60"
## [1] "0.115300456089227  , iter no is  61"
## [1] "0.116898442188084  , iter no is  62"
## [1] "0.1155482762492  , iter no is  63"
## [1] "0.117146892948912  , iter no is  64"
## [1] "0.115781465481869  , iter no is  65"
## [1] "0.117380764608197  , iter no is  66"
## [1] "0.116001064538696  , iter no is  67"
## [1] "0.117601084631379  , iter no is  68"
## [1] "0.116208021464217  , iter no is  69"
## [1] "0.117808790106229  , iter no is  70"
## [1] "0.116403202060688  , iter no is  71"
## [1] "0.118004737776041  , iter no is  72"
## [1] "0.116587398837861  , iter no is  73"
## [1] "0.118189712651817  , iter no is  74"
## [1] "0.116761338733414  , iter no is  75"
## [1] "0.118364435464754  , iter no is  76"
## [1] "0.116925689821128  , iter no is  77"
## [1] "0.118529569159169  , iter no is  78"
## [1] "0.117081067175074  , iter no is  79"
## [1] "0.118685724581677  , iter no is  80"
## [1] "0.117228038022064  , iter no is  81"
## [1] "0.118833465489964  , iter no is  82"
## [1] "0.117367126288529  , iter no is  83"
## [1] "0.11897331298073  , iter no is  84"
## [1] "0.117498816627662  , iter no is  85"
## [1] "0.119105749417662  , iter no is  86"
## [1] "0.117623557998069  , iter no is  87"
## [1] "0.119231221926906  , iter no is  88"
## [1] "0.117741766852765  , iter no is  89"
## [1] "0.119350145516007  , iter no is  90"
## [1] "0.117853829988522  , iter no is  91"
## [1] "0.119462905864035  , iter no is  92"
## [1] "0.117960107097891  , iter no is  93"
## [1] "0.11956986182331  , iter no is  94"
## [1] "0.118060933059762  , iter no is  95"
## [1] "0.119671347667201  , iter no is  96"
## [1] "0.118156619999713  , iter no is  97"
## [1] "0.119767675114137  , iter no is  98"
## [1] "0.118247459147157  , iter no is  99"
## [1] "Total iteration is  100"
```




```
## [1] "Estimated sigma is  0.447 ; True sigma is  1"
## [1] "Estimated phi is  9.49 ; True phi is  10"
## [1] "Estimated tau is  0.02798 ; True tau is  0.05"
## [1] "Estimated alpha is  0.43 ; True alpha is  1"
```

2.3 Kriging

We will now use the estimated model parameters to perform kriging prediction. Predict variables $x(s)$, where predictions sites lie on a regular grid of size 25x25 for the unit square. Visualize the Kriging surface and the prediction standard error. Compare with the true field.




```
## [1] "The prediction error is  0.977946945233227"
```

Part III Integrated nested Laplace Approximations (INLA)

In the last part of this exercise, we explore the `R-INLA` package along two examples.

3.1 Simple Linear Regression

First, we analyse the ski jumping data set using a linear regression model, which can be phrased as Latent Gaussian model suitable for INLA. Therefore, we start with loading the INLA package and exploring the dataset.

The ski jumping data set contains 26 observations of measured lengths in ski jumping competitions (in meters) between the years 1961 and 2011.

In Figure 1 we depict the 26 observations given their year. We observe a clear (almost linear) trend in the measured jumping lengths to increase with the years.

Suitable for this model assumption on the data, we use linear regression approach for the statistical modelling of this data, where the years x_i are the covariates and the lengths y_i are the responses for $i = 1, \ldots, 26$:

$$\mathbb{E}[y_i] = \mu + \beta x_i, \quad \mathbb{V}\partial\backslash[y_i] = \tau^{-1}.$$

This can be posed as a latent Gaussian model suited for the INLA framework.

1. The response depends on the linear predictors η as $y|x, \theta = \Pi\pi(y_i|\eta_i, \tau)$ with Gaussian likelihood $\pi(y_i|\eta_i, \tau) \sim \mathcal{N}(\eta_i, \tau^{-2})$

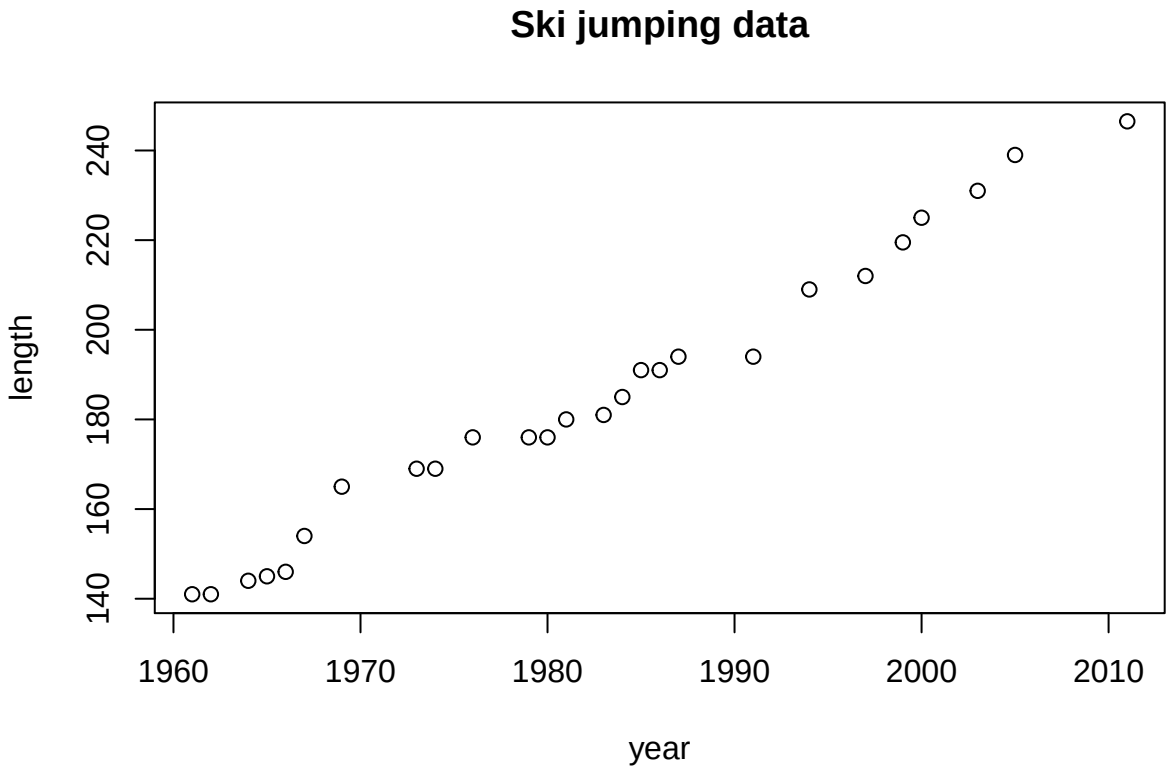


Figure 1: Visualisation of the ski jumping data set

2. The parameters μ and β of the linear predictor $eta_i = \mu + x_i\beta$ are independent Gaussian with a fixed huge variance and mean zero. Note that no additional hyperparameter is introduced here.
3. The model's hyperparameter τ is only one-dimensional and is equipped with a Gaussian prior by default with out specification.

The distributions which are not specified in detail here use default settings in the `R-INLA` package, which are naturally compatible with the LGM construction.

```
##
## Call:
##    "inla(formula = Length ~ Year, data = skiData, control.predictor =
##    list(compute = TRUE))"
## Time used:
##     Pre = 3.98, Running = 4.44, Post = 0.281, Total = 8.7
## Fixed effects:
##                   mean      sd 0.025quant  0.5quant 0.975quant      mode kld
## (Intercept) -4029.646 106.615  -4240.649 -4029.650  -3818.934 -4029.646   0
## Year            2.126   0.054      2.019     2.126      2.232     2.126   0
##
## Model hyperparameters:
##                                          mean   sd 0.025quant 0.5quant
## Precision for the Gaussian observations 0.072 0.02      0.038     0.07
##                                         0.975quant  mode
## Precision for the Gaussian observations      0.116 0.066
##
## Expected number of effective parameters(stdev): 2.00(0.00)
## Number of equivalent replicates : 13.00
```

```
##
## Marginal log-Likelihood:  -89.58
## Posterior marginals for the linear predictor and
##  the fitted values are computed
```

The INLA run generates posterior estimates for the fixed effects μ and β, which will be investigated blow. In the summary, we read the precision of the distribution for the hyperparameter τ which is rather small, meaning that the variance will be rather big. However, in a ski jumping competition we can expect a variance of several meters such that this is reasonable.

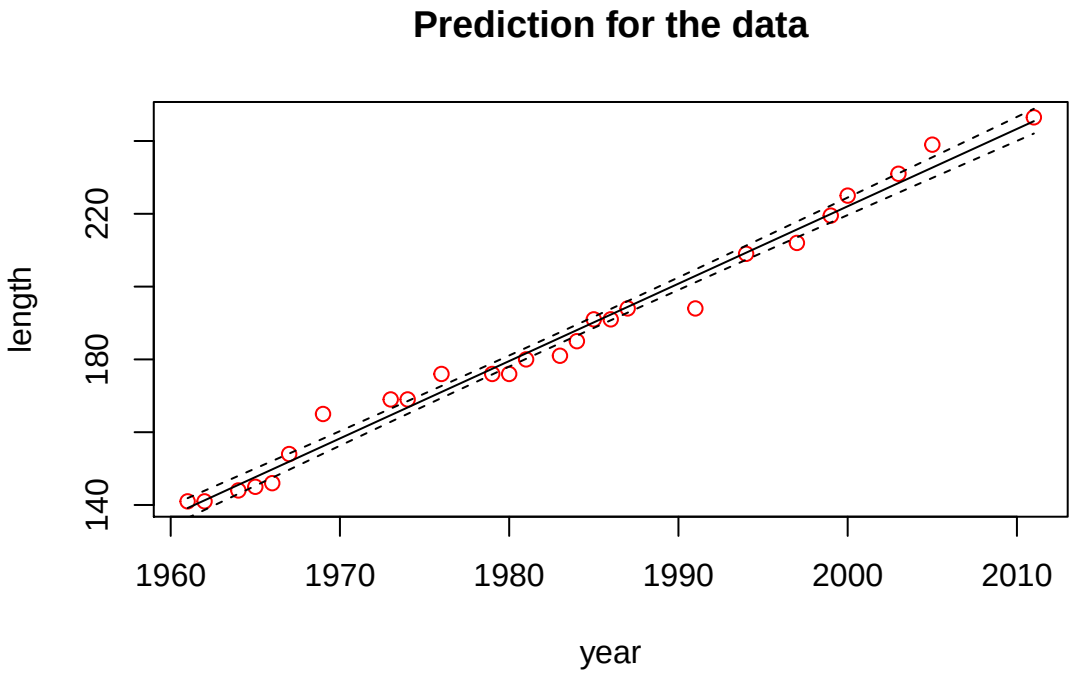


Figure 2: Linear regression with INLA

In Figure 2 we see that the linear trend is really well captured by the model (black line), whereas the 95% credibility interval (dashed line) does not cover all data points.

```
## [1] "Estimates for sigma:"
```

```
## Mean            3.83993
## Stdev           0.552976
## Quantile  0.025 2.93482
## Quantile  0.25  3.44617
## Quantile  0.5   3.77761
## Quantile  0.75  4.16357
## Quantile  0.975 5.09958
```

In Figure 3, we depict the marginal posterior distributions for all variables of interest. In particular, the transformed marginal analysis confirms our interpretation that the variance in the magnitude of several meters (roughly around 4 meters), but again this is very reasonable for this application.

3.2 GLMM with random effects

Last, we use INLA to analyse the "Seeds" data set. This data concerns the portion of seeds that germinated on a sample set of 21 plates. The plates are equipped with one of two types of seeds and one of two types of extracts. The characteristics of plate i are described by the covariates $x_{1,i}$ which is the type of seed and $x_{2,i}$

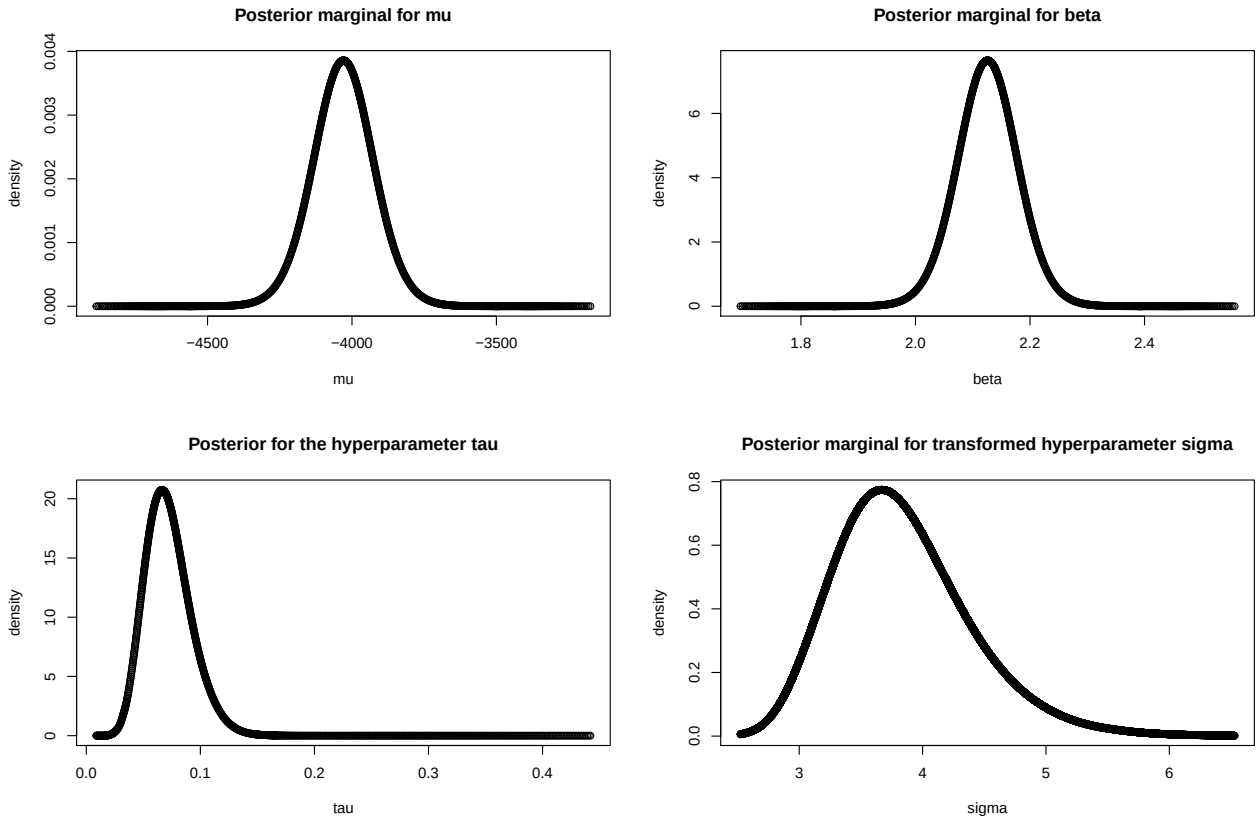


Figure 3: Posterior marginals for selected effects

which is the type of root extract - these covariates are either 0 or 1 for the different possibilies. Then the number of germinated seeds r_i on plate i is counted in contrast to the number of total seeds n_i on that plate. Having p_i as the probability of germination on plate i, a binomial model for this example is

$$r_i \sim Binomial(p_i, n_i)$$
$$\text{logit}(p_i) = a_0 + a_1 x_{1,i} + a_2 x_{2,i} + \varepsilon_i$$

where ε_i is some iid noise. As above non-specified prior- and hyperparameter-distributions use the default of `R-INLA`. This model is then again implemented in `R-INLA`.

```
## [1] "The x1 covariate:"

##  [1] 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1

## [1] "The x2 covariate:"

##  [1] 0 0 0 0 0 1 1 1 1 1 1 0 0 0 0 0 1 1 1 1 1

##
## Call:
##    c("inla(formula = formula, family = \"binomial\", data = data, Ntrials
##    = n, ", " control.predictor = list(compute = TRUE), control.family =
##    list(link = \"logit\"))" )
## Time used:
##     Pre = 6.14, Running = 0.708, Post = 0.38, Total = 7.23
## Fixed effects:
##               mean    sd 0.025quant 0.5quant 0.975quant   mode kld
## (Intercept) -0.429 0.115     -0.656   -0.429     -0.204 -0.428   0
## x1          -0.272 0.156     -0.580   -0.272      0.033 -0.271   0
## x2           1.066 0.146      0.782    1.066      1.353  1.065   0
##
## Random effects:
##   Name     Model
##     plate IID model
##
## Model hyperparameters:
##                           mean       sd 0.025quant 0.5quant 0.975quant mode
## Precision for plate 18930.69 20071.40      30.63 12701.76   73765.97 7.83
##
## Expected number of effective parameters(stdev): 3.25(1.16)
## Number of equivalent replicates : 6.46
##
## Marginal log-Likelihood:  -73.75
## Posterior marginals for the linear predictor and
##  the fitted values are computed
```

Remark the structured pattern in the covariates. Furthermore, here the precision is rather higher yielding a small variance in the estimate.

In Figure 4, we see rather confident estimates for the fixed effects of the model. In particular, we note that the effect a_2 will dominate the model since we only have 0 and 1 values for the covariates. The effect a_0 is common for all plates and the absolute value of a_1 for the seed type is a way smaller than the one for the root extract.

In Figure 5, we recognize a clear pattern with high values when the root extract with value 1 is chosen. The choice of the seed only has minor influence on the prediction. Moreover, the variance of this model cannot cover all data points, which is a common issue in binomial models with few data though.

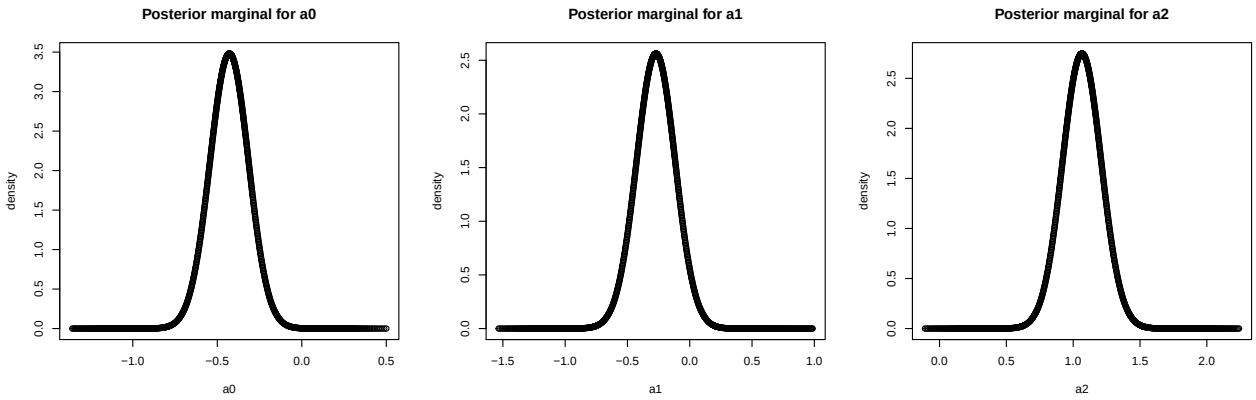


Figure 4: Posterior marginals for selected effects

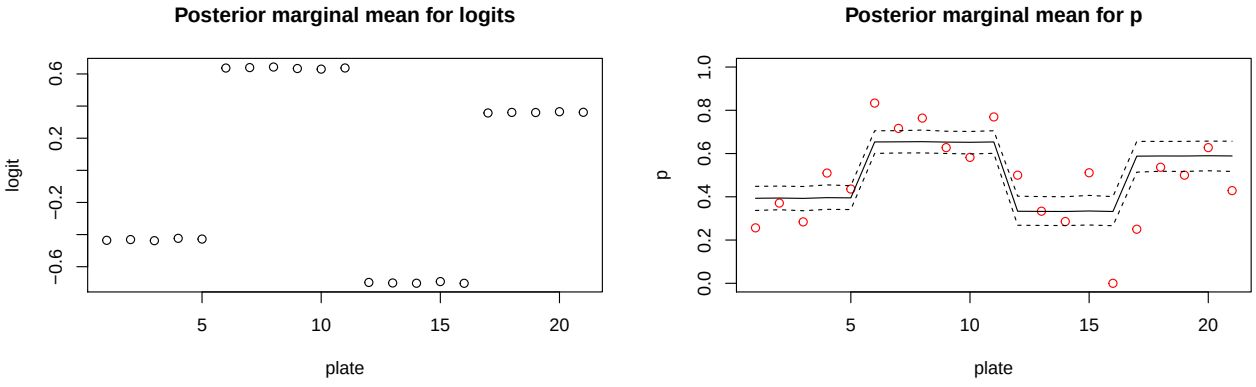


Figure 5: GLMM with INLA