Bayesian Data Analysis

Daniel Paulin & Nicolò Margaritella

University of Edinburgh

Semester 2, 2020/2021

With thanks to Jonathan Gair, Rubén Amorós-Salvador, Ken Newman, Vanda Inácio and Natalia Bochkina for much of the material

Outline

Section 1

Advice on Performing Analyses and Writing Reports

Hierarchical GLMs

Advice on Bayesian Analysis

Here are some general guidelines on carrying out Bayesian analysis (that don't always have to be followed!)

1. Be able to clearly and succinctly **state your objective**: What is it that you are trying to do? E.g., I want to see if water temperature or fish size affects survival rates.
2. Carry out **exploratory data analysis before** any model fitting. Informative plots (histograms, boxplots, and scatterplots) and numerical summaries.
3. Fit simple models, examine model **performance** (goodness of fit, predictive performance) and how **sensible parameter estimates** are (E.g., I expected survival to go down as the water got warmer but the model predicts that survival is going up—that does not make sense.). Then fit more complex models as needed or desired.
4. When using MCMC: always carry out **convergence checks**.
5. Examine the effect of your **priors**: try different ones.
6. Examine **predictive performance** of your final model.
7. **Honestly report** uncertainty, assumptions, concerns.

Hierarchical GLMs

Advice on Writing Reports of a Data Analysis

When you have analyzed a data set and are producing a written report of your analysis, some of the following remarks may be helpful. They are not absolute rules, but are often useful.

1. If a Table or Figure is included in the write-up, **label** it with a number, include a **caption**, and make **reference** to it in the writeup.

   Label & caption: Figure 1. Scatterplot of the estimated number of gray whales against the sea surface temperature (years 1990-2020).

   In the text: As shown in Figure 1, there is an apparent slightly positive relationship between sea surface temperature and the estimated number of gray whales.

Hierarchical GLMs

Advice on Writing Reports of a Data Analysis

2. When referring to a coefficient in a model, try to put its **numerical meaning into** the **context** of the data.
   - In the model predicting estimates of whale abundance as a function of temperature, the posterior distribution for the temperature coefficient is well above zero suggesting a relatively strong positive relationship between temperature and abundance. In particular, the model estimates that, given an increase of 10°C, the mean abundance of whales is multiplied by 1.3 (increase of 30%).

Hierarchical GLMs

Advice on Writing Reports of a Data Analysis

3. When writing a report do not expect the reader to have to figure out **what is important** and what is not. Tell them what you've learned.
   - Of the five environmental covariates that we examined, only water temperature at time of release appeared to have any relationship with salmon return rates. While others have found size at release important (reference ...), we did not find evidence for its importance in this study.

Hierarchical GLMs

Advice on Writing Reports of a Data Analysis

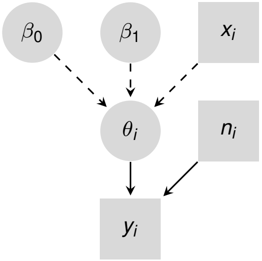
4. Briefly report **the technical details of the inference**, including type of algorithm, software and libraries, number of iterations and convergency checks. If computational time is to be discussed, include CPU, operative system, etc. This usually can be reported once for all the analyses.
   - Inference for all the models was carried out using Markov Chain Monte Carlo (MCMC) methods using the software packages `JAGS` and `R` and the library `rjags` package.
   - Three MCMC chains of 10 000 simulations each were obtained for each of the models after a burn-in of 5 000 iterations except for Model 2 which required 15 000 burn-in iterations.
   - Convergency of the MCMC simulation methods was assessed by direct inspection of the mixing of the chains and by the Gelman-Rubin-Brooks diagnostic (lower than 1.1 for all the traced parameters). The number of effective iterations was over 5 000 for all the parameters and the ratio between their standard deviation and the MCMC error was always greater than 20.

Hierarchical GLMs

Advice on Writing Reports of a Data Analysis

5. Write the **mathematical expression of the models** (usually in terms of conditional distributions). Optionally, you can include a directed graph for complex models [1].
   - The Binomial full model for the Beetles and its directed acyclic graph (DAG) representation:

$$
\begin{aligned}
y_i \mid \theta_i, \mathbf{x}_i &\sim \mathrm{Bin}(\theta_i, n_i), && i \in \{1, \ldots, I\} \\
\mathrm{logit}(\theta_i) &= \beta_0 + \beta_1 X_i \\
\beta_j &\sim \mathrm{N}(0, \sigma^2_{\beta_j}), && j \in \{0, 1\}
\end{aligned}
$$



We have chosen $\sigma_{\beta_j} = 100$ to reflect our lack of knowledge on the parameters β_j.

[1] Check file `DAG.tex` on Learn for a LaTeX example.

Section 2

Hierarchical GLMs

Hierarchical GLMs

Review

Recall the General Structure of a Hierarchical Model: two key ideas are

- ↪ Multiple levels in the model, a hierarchy, defined in terms of conditional probability distributions.
- ↪ Some parameter(s) different for each group of individuals of a certain partition, but with a common distribution (random effect).

A usual hierarchical model has:

- ↪ A **distribution for the observations** $y_{j,i}$, conditional on random parameters θ_j and covariates $x_{j,i}$:

$$y_{j,i}|\theta_j, x_{j,i} \quad \sim \quad \text{Distribution}\left(\theta_j, x_{j,i}\right) j = 1, \ldots, J, \quad i = 1, \ldots, n_j \tag{1}$$

- ↪ The **distribution for the random parameters** θ_j with unknown parameter(s) ψ:

$$\theta_j|\psi \quad \sim \quad \text{Distribution}\left(\psi\right), j = 1, \ldots, J \tag{2}$$

- ↪ Then a **(hyper-)prior distribution for the priors** with known hyperparameter(s) η:

$$\psi|\eta \quad \sim \quad \text{Distribution}\left(\eta\right) \tag{3}$$

Hierarchical GLMs

Introduction

Random effects can be included in generalised linear models, in what can be called hierarchical GLMs.

- $\hookrightarrow$ Allows the consideration of both fixed and random effects for non-Gaussian response variables.
- $\hookrightarrow$ The parameters of the random effects can be more interpretable as they are modeling the expectation of the data.
- $\hookrightarrow$ We can include extra variance for those distributions where the variance is determined by the expected value (Binomial, Poisson, Exponential...).

Hierarchical GLMs

Hatcheries example

y	Number of recovered salmons on each hatchery
n	Number of previously released salmons on each hatchery
alength	Average length of the released salmons on each hatchery

```
load("salmons.Rdata")
salmons

##     y   n alength
## 1   8  86    14.1
## 2  12  83    15.9
## 3  10  71    15.0
## 4   8  60    16.2
## 5  16  96    17.3
## 6  14  85    21.6
## 7  10  58    15.1
## 8  24  95    16.8
## 9  12  63    17.4
## 10 17 100    22.0
```

Hierarchical GLMs

No covariates (Hatcheries example)

We can model more interpretable random effects (and avoid colinearity problems).

$$Y_j \sim \mathrm{Bin}(n_j, p_j)$$

From Lecture 5:

$$p_j \sim \mathrm{Beta}(\alpha, \beta)$$
$$\alpha \sim \mathrm{Lognorm}(\ln(8.1), 0.7)$$
$$\beta \sim \mathrm{Lognorm}(\ln(21.3), 0.7)$$

New!

$$\mathrm{logit}(p_j) = \nu_j$$
$$\nu_j \sim \mathrm{N}(\mu_\nu, \sigma_\nu^2)$$
$$\mu_\nu \sim \mathrm{N}(0, 10^2)$$
$$\sigma_\nu \sim \mathrm{Unif}(0, 10)$$

Hierarchical GLMs

No covariates (Hatcheries example)

↪ Model Block

```
salmonB.model <- "model{
  for(j in 1:J) {
    y[j] ~ dbin(p[j],n[j])
    logit(p[j])  <-  nu[j]
    nu[j] ~ dnorm(mu.nu,tau.nu)
  }

  # Priors
  mu.nu   ~ dnorm(0, tau.mu.nu)
  tau.nu    <- pow(sigma.nu, -2)
  sigma.nu ~ dunif(0, UB.sigma.nu)

  # Tracing the expected probabilities
  E.surv <- ilogit(mu.nu)
 } "
```

Hierarchical GLMs

No covariates (Hatcheries example)

↪ Data and Inits

```
salmonB.data <- list(J=10, n=salmons$n, y=salmons$y,
                     tau.mu.nu=0.01, UB.sigma.nu=10)
salmonB.inits <- function(){ list(mu.nu=rnorm(1, 0, 5),
                                  sigma.nu=runif(0,5)) }
```

↪ Call to JAGS:

```
salmonB.res.A <- jags.model(file=textConnection(salmonB.model),
                            data=salmonB.data, inits=salmonB.inits,
                            n.chains=3, quiet = TRUE)
update(salmonB.res.A,n.iter=5000)
salmonB.res.B <- coda.samples(salmonB.res.A,
          variable.names=c("mu.nu","sigma.nu","E.surv","p"),
          n.iter=50000)
mcmcplots::mcmcplot(salmonB.res.B,
                    parms = c("mu.nu","sigma.nu","E.surv"))
```

Hierarchical GLMs

No covariates (Hatcheries example)

↪ Summary statistics

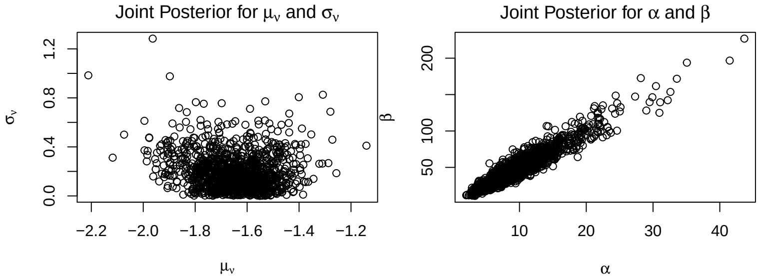
```
summary(salmonB.res.B)
# ...
#               Mean      SD  Naive SE Time-series SE
# mu.nu     -1.6446 0.12524 3.234e-04      0.0011650
# sigma.nu   0.2029 0.14277 3.686e-04      0.0022143
# E.surv     0.1626 0.01695 4.377e-05      0.0001584
# ...
```

```
summary(salmon.res.B)
# ...
#                    Mean        SD  Naive SE Time-series SE
# alpha            9.5578   4.38067 8.761e-03      6.834e-02
# beta            47.8301  22.34306 4.469e-02      3.489e-01
# expect.survival  0.1682   0.02141 4.281e-05      7.209e-05
# ...
```

Hierarchical GLMs

No covariates (Hatcheries example)

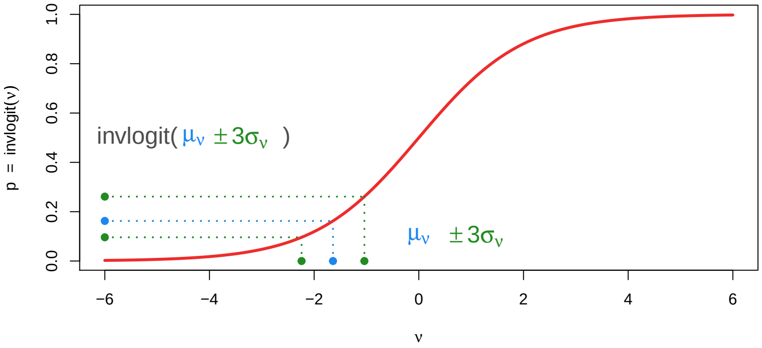
↪ The parameters μ_p and σ_p are much less correlated than α and β.



Hierarchical GLMs

No covariates (Hatcheries example)

$\hookrightarrow$ The parameters are more interpretable



Hierarchical GLMs

No covariates (Hatcheries example)

- The model can be rewriten so that one of the terms expresses an extra variance [2],
- by expressing the random effect ν_j as the sum of an overal mean $\mu = \mu_\nu$ plus a white noise term ϵ_j which follows a Gaussian distribution with zero mean and variance $\sigma^2_\varepsilon = \sigma^2_\nu$.

$$Y_j \sim \mathrm{Bin}(n_j, p_j)$$

$$\begin{aligned} \mathrm{logit}(p_j) &= \nu_j \\ \nu_j &\sim \mathrm{N}(\mu_\nu, \sigma^2_\nu) \\ \mu_\nu &\sim \mathrm{N}(0, 10^2) \\ \sigma_\nu &\sim \mathrm{Unif}(0, 10) \end{aligned} \qquad\qquad \begin{aligned} \mathrm{logit}(p_j) &= \mu + \varepsilon_j \\ \varepsilon_j &\sim \mathrm{N}(0, \sigma^2_\varepsilon) \\ \mu &\sim \mathrm{N}(0, 10^2) \\ \sigma_\varepsilon &\sim \mathrm{Unif}(0, 10) \end{aligned}$$

[2] If there were no extra variance term, the variance would be determined by the mean $Y_j \sim \mathrm{Bin}(p_j, n_j) \to E(Y_j) = n_j p_j, V(p_j) = n_j p_j (1 - p_j)$

Hierarchical GLMs

No covariates (Hatcheries example)

↪ The code (equivalent to the previous one but for names)

```
salmonB2.model <- "model{
  for(j in 1:J) {
    y[j] ~ dbin(p[j],n[j])
    logit(p[j])  <-  mu + epsilon[j]
    epsilon[j] ~ dnorm(0 ,tau.epsilon)
  }

  # Priors
  mu  ~ dnorm(0, tau.mu)
  tau.epsilon   <- pow(sigma.epsilon, -2)
  sigma.epsilon ~ dunif(0, UB.sigma.epsilon)

  # Tracing the expected probabilities
  E.surv <- ilogit(mu)
 } "
```

Hierarchical GLMs

Including covariates (Hatcheries example)

Let us include a covariate in the model.

$$
\begin{aligned}
Y_j &\sim \mathrm{Bin}(n_j, p_j) \\
\mathrm{logit}(p_j) &= \mu + \beta_1(X_j - \overline{X}) + \varepsilon_j \\
\mu &\sim \mathrm{N}(0, 10^2) \\
\beta_1 &\sim \mathrm{N}(0, 10^2) \\
\varepsilon_j &\sim \mathrm{N}(0, \sigma_\varepsilon^2) \\
\sigma_\varepsilon &\sim \mathrm{Unif}(0, 10)
\end{aligned}
$$

- $\hookrightarrow$ Traditionally the intercept of a regressor is called β_0.
- $\hookrightarrow$ In this case, the notation $\mu = \beta_0$ is adequate because we have centered our covariates and therefore μ is the expected value of the regressor for the expected value of X_j.

Hierarchical GLMs

Including covariates (Hatcheries example)

↪ The code - including centred covariates

```
salmonC.model <- "model{
  for(j in 1:J) {
    y[j] ~ dbin(p[j],n[j])
    logit(p[j])  <-  mu + beta1*(X[j] - mean(X[])) + epsilon[j]
    epsilon[j] ~ dnorm(0 ,tau.epsilon)
  }

  # Priors
  mu  ~ dnorm(0, tau.mu)
  beta1  ~ dnorm(0, tau.beta1)
  tau.epsilon   <- pow(sigma.epsilon, -2)
  sigma.epsilon ~ dunif(0, UB.sigma.epsilon)

  # Tracing the expected probabilities
  E.surv <- ilogit(mu)
 } "
```

Hierarchical GLMs

Including covariates (Hatcheries example)

↪ Data and Inits

```
salmonC.data <- list(J=10, n=salmons$n, y=salmons$y,
                     X=salmons$alength, tau.mu=0.01,
                     tau.beta1=0.01, UB.sigma.epsilon=10)
salmonC.inits <- function(){ list(mu=rnorm(1, 0, 2),
                                  beta1=rnorm(1, 0, 2),
                                  sigma.epsilon=runif(0,2)) }
```

↪ Call to JAGS:

```
salmonC.res.A <- jags.model(file=textConnection(salmonC.model),
                            data=salmonC.data, inits=salmonC.inits,
                            n.chains=3, quiet = TRUE)
update(salmonC.res.A,n.iter=5000)
salmonC.res.B <- coda.samples(salmonC.res.A,
          variable.names=c("mu","beta1","sigma.epsilon","E.surv","p")
          n.iter=50000)
mcmcplots::mcmcplot(salmonC.res.B,
                    parms = c("mu","beta1","sigma.epsilon","E.surv"))
```

Hierarchical GLMs

No covariates (Hatcheries example)

↪ Summary statistics

```
summary(salmonB.res.B)
# ...
#                Mean       SD  Naive SE Time-series SE
# mu.nu     -1.6446 0.12524 3.234e-04      0.0011650
# sigma.nu   0.2029 0.14277 3.686e-04      0.0022143
# E.surv     0.1626 0.01695 4.377e-05      0.0001584
# ...
```

```
summary(salmonC.res.B)
# ...
#                        Mean       SD  Naive SE Time-series SE
# mu              -1.65582 0.12985 3.353e-04      8.279e-04
# beta1            0.03521 0.04989 1.288e-04      3.356e-04
# sigma.epsilon    0.008831  0.094939  0.19137  0.30305  0.6047
# E.surv           0.16109 0.01745 4.504e-05      1.101e-04
# ...
```

Section 3

Posterior Predictive Distributions

Hierarchical GLMs

Posterior Predictive Distributions

- The posterior predictive distribution can be obtained by the law of total probability:

$$p(z|\boldsymbol{Y}) = \int_\Theta p(z|\theta, \boldsymbol{Y})p(\theta|\boldsymbol{Y})d\theta$$
$$= \int_\Theta p(z|\theta)p(\theta|\boldsymbol{Y})d\theta$$

- And in a hierarchical model, also by the same law:

$$p(\theta_{J+1}|\boldsymbol{Y}) = \int_\Psi p(\theta_{J+1}|\psi, \boldsymbol{Y})p(\psi|\boldsymbol{Y})d\psi$$
$$= \int_\Psi p(\theta_{J+1}|\psi)p(\psi|\boldsymbol{Y})d\psi$$

- For a variable (parameter) ϕ the samples obtained by MCMC $\phi^{(s)}$ are samples of $p(\phi|\boldsymbol{Y})$.
- Samples of a distribution obtained by an integral of the type

$$p(\gamma|\boldsymbol{Y}) = \int_\Phi p(\gamma|\phi)p(\phi|\boldsymbol{Y})d\phi$$

  can be obtained by simply substituting each $\phi^{(s)}$ in $p(\gamma|\phi)$ and sampling.
- Therefore, $\gamma^{(s)} = \mathit{sample}(p(\gamma|\phi^{(s)}))$ is a sample of $p(\gamma|\boldsymbol{Y})$.

Hierarchical GLMs

Posterior Predictive Distributions

- The posterior predictive distribution for an observation of group j, considering the random effect for that group, is

$$p(z_j|\boldsymbol{Y}) = \int_{\Theta_j} p(z|\theta_j)p(\theta_j|\boldsymbol{Y})d\theta_j$$

- As we have MCMC samples $\theta_j^{(s)}$ of $p(\theta_j|\boldsymbol{Y})$, we only have to draw a sample from $p(z|\theta_j^{(s)})$ for each $s \in \{1, ..., S\}$.
- But the posterior predictive distribution for an observation from a new group (or if we want to ignore the group) is

$$p(z_{J+1}|\boldsymbol{Y}) = \int_{\Theta_{J+1}} p(z_{J+1}|\theta_{J+1})p(\theta_{J+1}|\boldsymbol{Y})d\theta_{J+1}$$

- We do not have MCMC samples of $p(\theta_{J+1}|\boldsymbol{Y})$, so we obtain them from

$$p(\theta_{J+1}|\boldsymbol{Y}) = \int_{\Psi} p(\theta_{J+1}|\psi)p(\psi|\boldsymbol{Y})d\psi$$

by drawing a sample $\theta_{J+1}^{(s)}$ from $p(\theta_{J+1}|\psi^{(s)})$ and then a sample $z_{J+1}^{(s)}$ from $p(z_{J+1}|\theta_{J+1}^{(s)})$

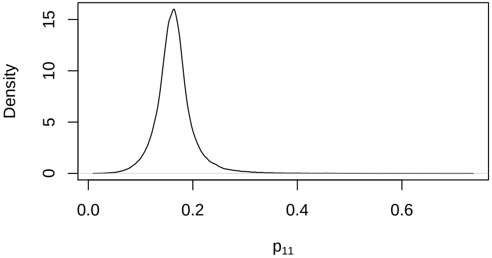
Hierarchical GLMs

Posterior Predictive Distributions

↪ To obtain samples from the posterior of a random effect for a new group $J+1$, we have to draw a sample from $p(\theta|\psi^{(s)})$ for each $s \in \{1, ..., S\}$. Draws from $p(p_{11}|\boldsymbol{y})$:

```
# Samples in data.frame form
salB.out <- do.call(rbind.data.frame, salmonB.res.B)
S <- dim(salB.out)[1]  # Num. MCMC samples

nu11 <- rnorm(n = S, mean = salB.out$mu.nu ,sd = salB.out$sigma.nu)
p11 <- 1/(1+ exp(-nu11) ) #inverse-logit
```



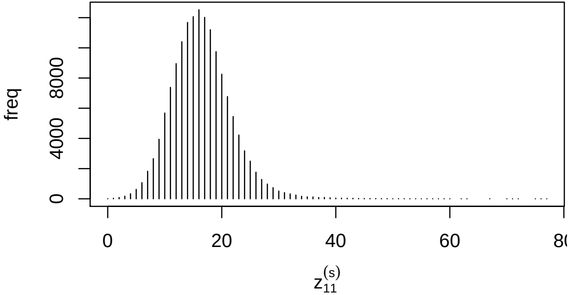
$$
\begin{aligned}
Y_j &\sim \mathrm{Bin}(n_j, p_j) \\
\mathrm{logit}(p_j) &= \nu_j \\
\nu_j &\sim \mathrm{N}(\mu_\nu, \sigma_\nu^2) \\
\mu_\nu &\sim \mathrm{N}(0, 10^2) \\
\sigma_\nu &\sim \mathrm{Unif}(0, 10)
\end{aligned}
$$

Hierarchical GLMs

Posterior Predictive Distributions

$\hookrightarrow$ Draws from $p(z_{11}|\boldsymbol{Y})$ (posterior predictive number of recoveries for a new hatchery if it releases 100 juvenile salmons)

```
nu11 <- rnorm(n = S, mean = salB.out$mu.nu ,sd = salB.out$sigma.nu)
p11 <- 1/(1+ exp(-nu11) )  # inverse-logit
z11 <- rbinom(n = S, size = 100, prob = p11)
```



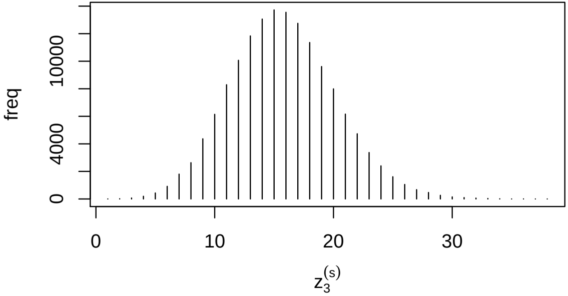
$$
\begin{aligned}
Y_j &\sim \mathrm{Bin}(n_j, p_j) \\
\mathrm{logit}(p_j) &= \nu_j \\
\nu_j &\sim \mathrm{N}(\mu_\nu, \sigma_\nu^2) \\
\mu_\nu &\sim \mathrm{N}(0, 10^2) \\
\sigma_\nu &\sim \mathrm{Unif}(0, 10)
\end{aligned}
$$

Hierarchical GLMs

Posterior Predictive Distributions

↪ Draws from $p(z_3|\boldsymbol{Y})$ (posterior predictive number of recoveries for the specific hatchery 3 it we releases 100 juvenile salmons, considering the particular value of the random effect)

```
z3 <- rbinom(n = S, size = 100, prob = salB.out$`p[3]`)
```



$$
\begin{aligned}
Y_j &\sim \text{Bin}(n_j, p_j)\\
\text{logit}(p_j) &= \nu_j\\
\nu_j &\sim \text{N}(\mu_\nu, \sigma_\nu^2)\\
\mu_\nu &\sim \text{N}(0, 10^2)\\
\sigma_\nu &\sim \text{Unif}(0, 10)
\end{aligned}
$$

Hierarchical GLMs

Posterior Predictive Distributions

Some further remarks:

- ↪ If there are **covariates**, they **should be taken into account**. Are we predicting for a certain value of X? For the mean value?
- ↪ Posterior and posterior predictive distributions can also be done **in JAGS** (as we have seen before). Be cautious when you predict for $p(z_j|\boldsymbol{Y})$. The prediction is different if you consider the particular random effect for j or not.

Section 4

Model Selection

Hierarchical GLMs

Model Selection

Bayesian Model Selection.

- Given two or more competing models how should one decide which model is better?
- Which model predicts future observations better is one approach.
- Cross-validation is often a good approach: leave out a portion of the data (test set), and fit competing models to the rest of the data (training set). Then see which model predicts the test set best.
- Which model has more sensible coefficients is another.
- Special priors for the parameters of the regression can also be used for variable selection (e.g. Laplace priors –Bayesian Lasso– or Spike-and-slab priors).
- Other approaches are based on setting a prior distribution to each model and compare the Bayes Factor

$$\frac{Pr(\boldsymbol{Y}|M_1)}{Pr(\boldsymbol{Y}|M_2)} = \frac{Pr(M_1|\boldsymbol{Y})Pr(M_1)}{Pr(M_2|\boldsymbol{Y})Pr(M_2)}$$

Hierarchical GLMs

Model Selection

- Large datasets and complex models hinder the use of cross validation and Bayes factor. Model-fit criteria represent a good alternative.

  Deviance Information Crieterion (DIC).

- DIC provides a (1) measure of the error in the fit with (2) a penalty for the complexity of the model.
- The lower DIC, the better the model.
- (1) The measure of error is called the "Deviance" and it is 2 times the negative log likelihood:

$$\text{Deviance:} \quad D(\mathbf{Y}|\theta) \quad = -2 \ln \mathrm{f}(\mathbf{Y}|\theta),$$

  The smaller the deviance, "generally speaking", the better the model fits the data.

- The deviance $D(\mathbf{Y}|\theta)$ is a random variable in the Bayesian setting (as θ is random) with its own posterior distribution

Hierarchical GLMs

Model Selection

Deviance Information Crieterion (DIC).

- (2) Model complexity is summarized by the *Effective Number of Parameters*, pD.
- pD is similar to a count of the unknown parameters in a model where highly dependent parameters and those with strongly informative priors count for less than one. Particularly in hierarchical models, the effective number of parameters does not equal the number of parameters.

$$p_D = \widehat{D(\mathbf{Y},\theta)} - D(\mathbf{Y},\bar{\theta})$$

where $\widehat{D(\mathbf{Y},\theta)}$ is the estimate of the posterior mean of the deviance which can be obtained from

$$\widehat{D(\mathbf{Y},\theta)} \approx \frac{1}{S}\sum_{s=1}^{S} D(\mathbf{y},\theta^{(s)})$$

where $\theta^{(s)}$ is the sth sample from the MCMC chain (after burn-in).

- $D(\mathbf{Y},\bar{\theta})$ is the deviance evaluated at the posterior mean of θ.

Hierarchical GLMs

Model Selection

- DIC is then the sum of (1) and (2):

$$DIC = \widehat{D(\mathbf{y}, \theta)} + p_D$$

- The actual value of the DIC is hard to interpret; however, it can be employed to rank models: models with small DIC are simple (small pD) and fit well the data (small Deviance).

- Loose rule of thumb: Difference of 5= substantial; 10 = definitive.

- Note that pD can be outside the range [0; p] in some pathological cases, typically where the posterior mean of θ is not a good summary of the posterior (e.g. mixture models with multimodal priors and posteriors).

Hierarchical GLMs

Model Selection

Getting DIC from JAGS (our Salmon example: model with vs. without covariate):

```
salmonB.DIC <- dic.samples(model=salmonB.res.A,n.iter=10000,type="pD")
salmonC.DIC <- dic.samples(model=salmonB.res.A,n.iter=10000,type="pD")
```

```
salmonB.DIC  # Model without average length

## Mean deviance:  50.73
## penalty 4.113
## Penalized deviance: 54.84

salmonC.DIC  # Model with average length

## Mean deviance:  50.69
## penalty 4.036
## Penalized deviance: 54.73
```

Section 5

Spatial and Temporal Modelling in R-INLA

Spatial and Temporal Modelling

Introduction to the dataset

- In this section, we are going to show you how can random effects be used for spatial and temporal modelling.
- R-INLA will be used to perform the calculations.
- The dataset considered is the apartment price data from Taipei that we used previously in Assignment 1. This dataset contains 414 rows. The response variable is

$$y = \log(\text{house price per unit area}).$$

The following covariates are available:

- **transaction date** (this is stored as a real number between 2012 and 2014, i.e. 2013.5 corresponds to 1 July, 2013)
- **house age** (in years)
- **distance from nearest MRT (metro) station** (in meters)
- **number of convenience stores within walking distance**
- **latitude** (in degrees)
- **longitude** (in degrees)

Spatial and Temporal Modelling

Introduction to the dataset

```
house <- read.csv(file = 'Real_estate.csv')
head(house)
```

```
##   No X1.transaction.date X2.house.age X3.distance.to.the.nearest.MRT.station
## 1  1            2012.917         32.0                               84.87882
## 2  2            2012.917         19.5                              306.59470
## 3  3            2013.583         13.3                              561.98450
## 4  4            2013.500         13.3                              561.98450
## 5  5            2012.833          5.0                              390.56840
## 6  6            2012.667          7.1                             2175.03000
##   X4.number.of.convenience.stores X5.latitude X6.longitude
## 1                              10    24.98298     121.5402
## 2                               9    24.98034     121.5395
## 3                               5    24.98746     121.5439
## 4                               5    24.98746     121.5439
## 5                               5    24.97937     121.5425
## 6                               3    24.96305     121.5125
##   Y.house.price.of.unit.area
## 1                       37.9
## 2                       42.2
## 3                       47.3
## 4                       54.8
## 5                       43.1
## 6                       32.1
```

Spatial and Temporal Modelling

Overview of Taipei

Spatial and Temporal Modelling I

INLA model 1: linear model with only fixed effects

- We start by creating a baseline Bayesian linear regression model with only fixed effects.
- The first step is to standardize all of the covariates, and create the response variable y as the logarithm of the house price per unit area.
- After this, we fit a model using the following code,

```
m.I <- inla(y ~ 1 + transaction + age + distance+ stores
            +longitude+latitude,
            family = "Gaussian",
            data = house,
            control.compute = list(cpo=T,dic = T))
```

Spatial and Temporal Modelling I

INLA model 1: linear model with only fixed effects

```
summary(m.I)
Call:
   c("inla(formula = y ~ 1 + transaction + age + distance + stores + ", "
   longitude + latitude, family = \"Gaussian\", data = house, ", "
   control.compute = list(cpo = T, dic = T))")
Time used:
    Pre = 3.33, Running = 0.376, Post = 0.294, Total = 3.99
Fixed effects:
              mean    sd 0.025quant 0.5quant 0.975quant   mode kld
(Intercept)  3.567 0.011      3.545    3.567      3.588  3.567   0
transaction  0.038 0.011      0.017    0.038      0.060  0.038   0
age         -0.079 0.011     -0.101   -0.079     -0.058 -0.079   0
distance    -0.184 0.023     -0.228   -0.184     -0.139 -0.184   0
stores       0.082 0.014      0.054    0.082      0.109  0.082   0
longitude    0.006 0.019     -0.031    0.006      0.042  0.006   0
latitude     0.098 0.014      0.071    0.098      0.126  0.098   0
```

Spatial and Temporal Modelling II

INLA model 1: linear model with only fixed effects

```
Model hyperparameters:
                                         mean   sd 0.025quant 0.5quant
Precision for the Gaussian observations 20.46 1.43      17.76    20.43
                                        0.975quant  mode
Precision for the Gaussian observations      23.36 20.36

Expected number of effective parameters(stdev): 7.05(0.004)
Number of equivalent replicates : 58.71

Deviance Information Criterion (DIC) ...............: -63.57
Deviance Information Criterion (DIC, saturated) ....: 425.59
Effective number of parameters .....................: 8.14

Marginal log-Likelihood:  -19.21
CPO and PIT are computed
Posterior marginals for the linear predictor and
 the fitted values are computed
```

Spatial and Temporal Modelling

INLA model 1: linear model with only fixed effects

- We also compute NLSCPO, DIC, and the standard deviation of mean residuals for INLA model 1.

```
## NLSCPO of INLA model 1: -28.70251
## DIC of INLA model 1: -63.5663
## Standard deviation of mean residuals for INLA model 1: 0.2199845
```

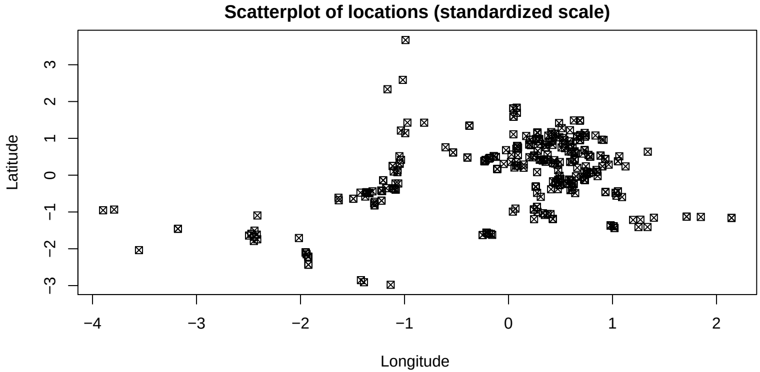
- As the response variable y is in log-scale, 0.22 standard error corresponds to approximately 22% relative error in the estimate of the house price (per unit area).
- This is quite large, and it raises the possibility that some important aspects that influence the house price are not included in the data, or not captured by our simple model.

Spatial and Temporal Modelling

INLA model 2: SPDE spatial random effects

$\hookrightarrow$ One obvious candidate for improved modelling is the location (latitude + longitude).

```
plot(house$longitude, house$latitude,
     main="Scatterplot of locations (standardized scale)",
     xlab="Longitude ", ylab="Latitude", pch=7)
```



Spatial and Temporal Modelling

INLA model 2: SPDE spatial random effects

- One way to use the location information is to model it with a smooth Gaussian field random effect.
- This means that there would a location specific effect on the response (log house price per unit area), but this effect is assumed to be changing relatively smoothly in the location (to make sure that we do not overfit on the noise).
- Stochastic Partial Differential Equation (SPDE) random effects were introduced by Finn Lindgren et al., “An explicit link between Gaussian fields and Gaussian Markov random fields: the stochastic partial differential equation approach.” JRSSB 73.4 (2011): 423-498.
- They allow the creation of efficient spatial random effect models where the latent variables are chosen according to appropriately chosen meshes.

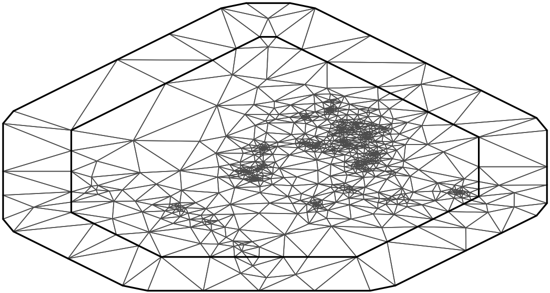
Spatial and Temporal Modelling

INLA model 2: SPDE spatial random effects

↪ When using spde spatial models in INLA, the first step is to create a mesh.

```
Locations = cbind(house$longitude, house$latitude)
loc.mesh <- inla.mesh.2d(Locations, max.edge = c(10, 20))
plot(loc.mesh)
```

Constrained refined Delaunay triangulation



Spatial and Temporal Modelling

INLA model 2: SPDE spatial random effects

- Once the mesh is ready, we need to create the A, spde and w objects that together define the SPDE random effects model. These can be created using the `inla.spde.make.A`, `inla.spde2.pcmatern` and `inla.spde.make.index` functions.

```
# Creating A matrix (Observation/prediction matrix)
loc.A <- inla.spde.make.A(loc.mesh, loc = Locations)
#Creating Matern SPDE model object with PC prior
loc.spde = inla.spde2.pcmatern(mesh = loc.mesh,
           prior.range = c(1, 0.5),
           prior.sigma = c(1, 0.5))
#Generating the SPDE model index vector
loc.w <- inla.spde.make.index('w', n.spde = loc.spde$n.spde)
```

- In the case of SPDE models, the dataframe that we pass along to INLA as the `data` argument changes (i.e. we cannot just use the house variable).
- We need to create the `data` argument using the `inla.stack` and `inla.stack.data` functions.

Spatial and Temporal Modelling

INLA model 2: SPDE spatial random effects

```
#First we make the model matrix using the model formula,
#but without response and intercept.
X0 <- model.matrix( ~ 0 +transaction+age+distance+ stores, data = house)
X <- as.data.frame(X0) # convert to a data frame.
# Making the stack
N <- nrow(house) #Saving the number of rows in the data
StackHouse <- inla.stack(
  # specify the response variable
  data = list(y = house$y),
  # Vector of Multiplication factors for  fixed effects
  A = list(1, 1, loc.A),
  #Specify the fixed and random effects
  effects = list(
    # specify the manual intercept!
    Intercept = rep(1, N),
    # attach the model matrix
    X = X,
    # attach the w
    w = loc.w) )
```

Spatial and Temporal Modelling

INLA model 2: SPDE spatial random effects

↪ The INLA call is described below.

```
m.I2 <- inla(y ~ 0 + Intercept + transaction + age + distance+ stores
             +f(w, model = loc.spde),
             family = "Gaussian",
             data = inla.stack.data(StackHouse),
             control.compute = list(cpo=T,dic = T),
             control.predictor = list(A = inla.stack.A(StackHouse)))
```

↪ `inla.stack.data(StackHouse)` is used to pass along the data argument.

↪ The SPDE random effect is denoted as `f(w, model = loc.spde)`.

```
summary(m.I2)
Call:
   c("inla(formula = y ~ 0 + Intercept + transaction + age + distance + "
   " stores + f(w, model = loc.spde), family = \"Gaussian\", data =
   inla.stack.data(StackHouse), ", " control.compute = list(cpo = T, dic
   T), control.predictor = list(A = inla.stack.A(StackHouse)))" )
Time used:
    Pre = 3.41, Running = 1.53, Post = 0.34, Total = 5.28
Fixed effects:
              mean    sd 0.025quant 0.5quant 0.975quant   mode kld
Intercept    3.537 0.085      3.369    3.536      3.711  3.535   0
transaction  0.044 0.008      0.027    0.044      0.060  0.044   0
age         -0.058 0.012     -0.081   -0.058     -0.035 -0.058   0
distance    -0.175 0.045     -0.267   -0.175     -0.086 -0.174   0
stores       0.068 0.035     -0.001    0.068      0.139  0.068   0

Random effects:
  Name    Model
    w SPDE2 model

Model hyperparameters:
```

```
                                         mean    sd 0.025quant 0.5quant
Precision for the Gaussian observations 44.992 5.486     34.792   44.824
Range for w                              0.878 0.255      0.508    0.833
Stdev for w                              0.274 0.036      0.210    0.272
                                        0.975quant   mode
Precision for the Gaussian observations     56.337 44.650
Range for w                                  1.499  0.748
Stdev for w                                  0.351  0.268

Expected number of effective parameters(stdev): 99.50(16.18)
Number of equivalent replicates : 4.16

Deviance Information Criterion (DIC) ...............: -296.27
Deviance Information Criterion (DIC, saturated) ....: 522.91
Effective number of parameters .....................: 102.01

Marginal log-Likelihood:  50.27
CPO and PIT are computed

Posterior marginals for the linear predictor and
 the fitted values are computed
```

Spatial and Temporal Modelling

INLA model 2: SPDE spatial random effects

↪ We also compute NLSCPO, DIC, and the standard deviation of mean residuals for INLA model 2.

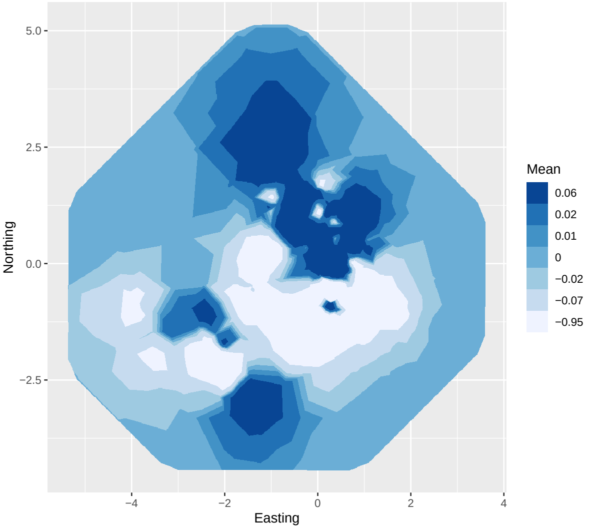
```
## NLSCPO of INLA model 2: -117.01
## DIC of INLA model 2: -307.2324
## Standard deviation of mean residuals for INLA model 2: 0.1247535
```

↪ Our second model is significantly better than the first one based on all 3 criteria. Using spatial effects have significantly increased the accuracy of the model.

↪ The figure on the next slide shows the mean of the spatial effects (plotted using `ggField`, see the R code Lecture 6 on Learn). We can see that some areas have higher house prices than others.

Spatial and Temporal Modelling

INLA model 2: SPDE spatial random effects



Spatial and Temporal Modelling

INLA model 3: SPDE spatial and temporal random effects

- ↪ In model 2, we have considered spatial random effects for the location parameter.
- ↪ In model 3, we consider random effects for the date and house age (temporal parameters) and distance to nearest MRT station (spatial parameter).
- ↪ All of these are one dimensional. We are going to create one dimensional meshes for them, and then use 3 additional SPDE random effects.
- ↪ The specification of these in R-INLA is quite similar to the previous spatial random effect.

Spatial and Temporal Modelling

INLA model 3: SPDE spatial and temporal random effects

```
Locations = cbind(house$longitude, house$latitude)
loc.mesh <- inla.mesh.2d(Locations, max.edge = c(10, 20))

loc.A <- inla.spde.make.A(loc.mesh, loc = Locations)
loc.spde = inla.spde2.pcmatern(mesh = loc.mesh,
                                prior.range = c(1, 0.5),
                                prior.sigma = c(1, 0.5))
loc.w <- inla.spde.make.index('w', n.spde = loc.spde$n.spde)

d.mesh <- inla.mesh.1d(house$distance) #Create a 1D mesh
d.A <- inla.spde.make.A(d.mesh, loc = house$distance)
d.spde = inla.spde2.pcmatern(mesh = d.mesh,
         prior.range = c(1, 0.5),
         prior.sigma = c(1, 0.5))
d.w <- inla.spde.make.index('d.w', n.spde = d.spde$n.spde)
```

Spatial and Temporal Modelling

INLA model 3: SPDE spatial and temporal random effects

```
a.mesh <- inla.mesh.1d(house$age)
a.A <- inla.spde.make.A(a.mesh, loc = house$age)
a.spde = inla.spde2.pcmatern(mesh = a.mesh,
         prior.range = c(1, 0.5),
         prior.sigma = c(1, 0.5))
a.w <- inla.spde.make.index('a.w', n.spde = a.spde$n.spde)

t.mesh <- inla.mesh.1d(seq(min(house$transaction),
          max(house$transaction),length.out=100))
t.A <- inla.spde.make.A(t.mesh, loc = house$transaction)
t.spde = inla.spde2.pcmatern(mesh = t.mesh,
         prior.range = c(1, 0.5),
         prior.sigma = c(1, 0.5))
t.w <- inla.spde.make.index('t.w', n.spde = t.spde$n.spde)
```

Spatial and Temporal Modelling

INLA model 3: SPDE spatial and temporal random effects

```
# Make the model matrix using the model formula without response and intercept.
X0 <- model.matrix( ~ 0 +transaction+age+distance+ stores, data = house)
X <- as.data.frame(X0) # convert to a data frame.
# Making the stack ####
N <- nrow(house)
house$ID=1:N
StackHouse <- inla.stack(
  data = list(y = house$y), # specify the response variable
  # Vector of Multiplication factors for fixed and random effects
  A = list(1, 1, a.A, t.A, d.A, loc.A),
  effects = list(
    Intercept = rep(1, N), # specify the manual intercept!
    X = X, # attach the model matrix
    a.w = a.w,
    t.w = t.w,
    d.w = d.w,
    w = loc.w # attach the w
    ) )
```

Spatial and Temporal Modelling

INLA model 3: SPDE spatial and temporal random effects

- The call to INLA for model 3:

```
m.I3 <- inla(y ~ 0 + Intercept + transaction + age + distance+ stores
          +f(a.w,model=a.spde)+f(t.w,model=t.spde)
          +f(d.w,model=d.spde)+f(w, model = loc.spde),
          family = "Gaussian",
          data = inla.stack.data(StackHouse),
          control.compute = list(cpo=T,dic = T),
          control.predictor = list(A = inla.stack.A(StackHouse)))
```

- The data parameter is still passed along as `inla.stack.data(StackHouse)`.
- The 4 SPDE random effects are specified as `f(a.w,model=a.spde)`, `f(t.w,model=t.spde)`, `f(d.w,model=d.spde)`, `f(w, model = loc.spde)`.

Spatial and Temporal Modelling I

INLA model 3: SPDE spatial and temporal random effects

```
summary(m.I3)
Call:
   c("inla(formula = y ~ 0 + Intercept + transaction + age + distance + "
   " stores + f(a.w, model = a.spde) + f(t.w, model = t.spde) + ", "
   f(d.w, model = d.spde) + f(w, model = loc.spde), family = \"Gaussian\"
   ", " data = inla.stack.data(StackHouse), control.compute = list(cpo =
   T, ", " dic = T), control.predictor = list(A =
   inla.stack.A(StackHouse))" )
Time used:
    Pre = 5.85, Running = 16.3, Post = 0.695, Total = 22.8
Fixed effects:
              mean    sd 0.025quant 0.5quant 0.975quant   mode   kld
Intercept    3.630 1.807     -0.643    3.644      7.819  3.654 0.000
transaction  0.051 0.037     -0.022    0.049      0.127  0.046 0.002
age         -0.019 0.095     -0.203   -0.020      0.168 -0.025 0.000
distance    -0.230 0.122     -0.471   -0.230      0.011 -0.230 0.000
stores       0.064 0.038     -0.010    0.063      0.137  0.063 0.000
```

Spatial and Temporal Modelling II

INLA model 3: SPDE spatial and temporal random effects

```
Random effects:
  Name    Model
    a.w SPDE2 model
   t.w SPDE2 model
   d.w SPDE2 model
   w SPDE2 model

Model hyperparameters:
                                          mean     sd 0.025quant 0.5quant
Precision for the Gaussian observations 51.341  4.674     42.583   51.212
Range for a.w                            6.843  3.762      2.390    5.930
Stdev for a.w                            0.400  0.321      0.106    0.307
Range for t.w                           28.315 35.514      4.967   17.683
Stdev for t.w                            0.456  0.486      0.062    0.311
Range for d.w                            4.754  4.024      1.018    3.597
Stdev for d.w                            0.083  0.083      0.001    0.051
```

Spatial and Temporal Modelling III

INLA model 3: SPDE spatial and temporal random effects

```
Range for w                                0.624  0.190      0.375    0.582
Stdev for w                                0.142  0.052      0.056    0.140
                                         0.975quant   mode
Precision for the Gaussian observations      60.913 51.053
Range for a.w                                16.574  4.573
Stdev for a.w                                 1.249  0.202
Range for t.w                               116.079  9.516
Stdev for t.w                                 1.731  0.156
Range for d.w                                15.421  2.244
Stdev for d.w                                 0.284  0.000
Range for w                                   1.101  0.503
Stdev for w                                   0.233  0.116

Expected number of effective parameters(stdev): 124.37(9.43)
Number of equivalent replicates : 3.33

Deviance Information Criterion (DIC) ...............: -339.23
```

Spatial and Temporal Modelling IV

INLA model 3: SPDE spatial and temporal random effects

```
Deviance Information Criterion (DIC, saturated) ....: 531.90
Effective number of parameters .....................: 122.70

Marginal log-Likelihood:  47.33
CPO and PIT are computed

Posterior marginals for the linear predictor and
 the fitted values are computed
```

Spatial and Temporal Modelling

INLA model 3: SPDE spatial and temporal random effects

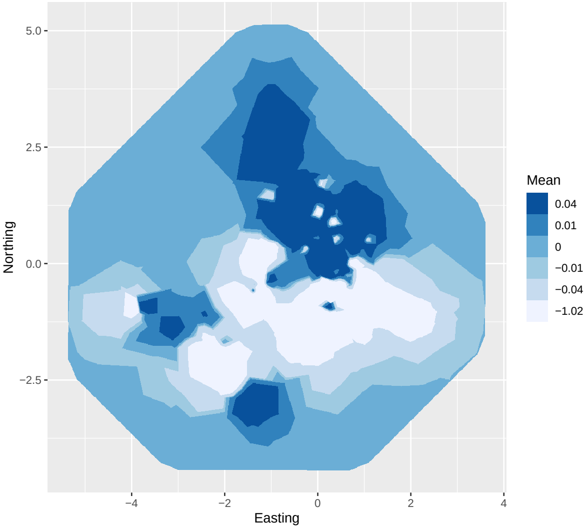
- We also compute NLSCPO, DIC, and the standard deviation of mean residuals for INLA model 3.

```
## NLSCPO of INLA model 3: -122.6155
## DIC of  INLA model 3: -339.2312
## Standard deviation of mean residuals for INLA model 3: 0.1160327
```

- We can see that this is the best model according to all 3 criteria. By using both spatial and temporal random effects, we have been able to descrease the standard deviation of the mean residuals by half!
- The figure on the next slide shows the mean of the spatial effects for INLA model 3.
- This is quite similar to what we had for model 2.

Spatial and Temporal Modelling

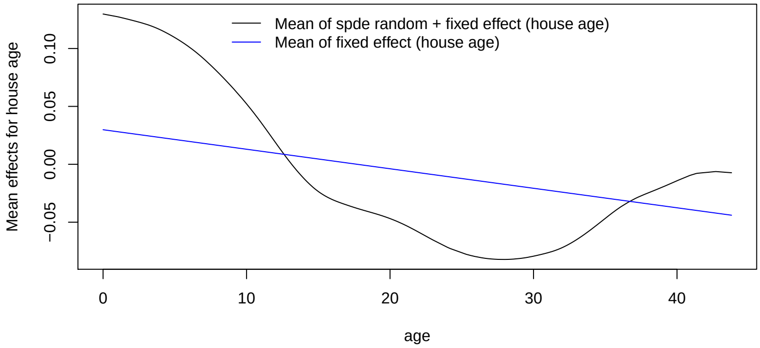
INLA model 3: SPDE spatial and temporal random effects



Spatial and Temporal Modelling

INLA model 3: SPDE spatial and temporal random effects

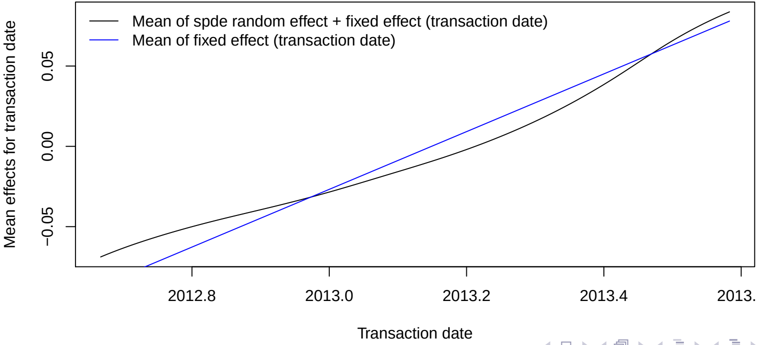
↪ As we can see on the plot, the dependence on the house age of the mean random effects is close to linear between 0 to 20 years, but becomes highly non-linear above 20 years.



Spatial and Temporal Modelling

INLA model 3: SPDE spatial and temporal random effects

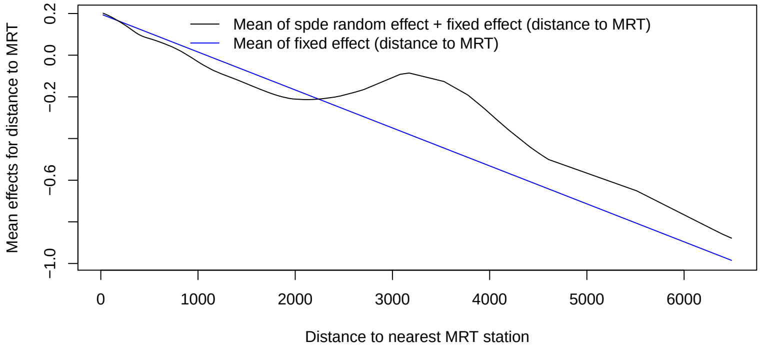
↪ As we can see on the plot, the dependence of the mean effect for transaction date on the transaction time is not very far from linear, but there is clearly faster growth starting from 2013.4, which corresponds to spring time. Indeed, spring and summer seems to be the best time to sell a house as the market is most active at that time, see `https://www.theadvisory.co.uk/house-selling/best-time-to-sell-house/`.



Spatial and Temporal Modelling

INLA model 3: SPDE spatial and temporal random effects

↪ The dependence of the mean effect for distance to MRT on the distance to MRT is not very far from linear, but there is a flat area between 2500-3500 meters..



Spatial and Temporal Modelling

Conclusion

- ↪ Overall, INLA Models 2 and 3 have much better performance than the simple Bayesian linear regression model, INLA Model 1.
- ↪ The advantage will further increase with more data, as these spde random effect models are able to flexibly increase their complexity (effective number of parameters) to match the amount of available data.
- ↪ The effective number of parameters is controlled by the number of cells in the mesh, and the priors of the spde models.
- ↪ The number of parameters of the linear regression model is fixed at 7 so it is unable to express complex spatial and temporal relationships.