A Primer for using INLA to Estimate Ground-Motion Models

Nicolas Kuehn

Version 1 - 06 September, 2021

Contents

1 Introduction

Empirical Ground-Motion Models (GMMs) are typically estimated by regression. While there is more to GMM development than fitting model to data, regression is an indispensable tool to connect the model coefficients to the data. One important aspect for regression in the context of GMM development is to account for correlations between records from the same event/station (or region for partially nonergodic models), which is typically done by estimating a random effect for these terms (Al-Atik et al. 2010). In the computer language **R** (https://www.r-project.org/), a popular package to estimate random effects models is **lme4** (Bates et al. 2015), which provides efficient algorithms to estimate these kinds of models via (restricted) maximum-likelihood (ML). **lme4** has been used extensively in empirical GMM development (e.g. Weatherill, Kotha, and Cotton 2020; Montalva, Bastías, and Rodriguez-Marek 2017; Bindi et al. 2018, 2019; Bajaj and Anbazhagan 2018; Campbell and Bozorgnia 2019; Stafford 2014; Kotha et al. 2020; Sahakian et al. 2018; Lanzano et al. 2019; Bora, Cotton, and Scherbaum 2019; Kotha, Cotton, and Bindi 2018; Bahrampouri, Rodriguez-Marek, and Green 2021; Parker et al. 2020). Similar packages/functions exist for other computer languages (such as `nlmefit` in Matlab (https://www.mathworks.com/), used in e.g Sedaghati and Pezeshk (2017) and Farajpour, Pezeshk, and Zare (2019)). Generally, one can probably say that ML-estimation remains the workhorse in empirical GMM development.

Contrary to ML-estimation, Bayesian inference has sometimes been used in GMM development (e.g. Wang and Takada 2009; Stafford 2019, 2014; Kuehn and Scherbaum 2015, 2016; Kuehn and Abrahamson 2018, 2020; Kuehn, Abrahamson, and Walling 2019; Rahpeyma et al. 2018; Kowsari et al. 2019, 2020; Ordaz, Arciniega, and Singh 1994; Arroyo and Ordaz 2010a, 2010b). In the end, GMMs can be successfully estimated using ML or Bayesian inference, and it can be a matter of convenience and familiarity which tool is chosen. I want to note that GMM development requires a lot of judgment to make sure that the estimated GMM is physically viable. Such judgment can be incorporated into a Bayesian model via the prior distributions for the parameters. In addition, the posterior distribution provides an easy tool to probabilistically assess the (within-model) epistemic uncertainty associated with model predictions. Stafford (2019) also noted that GMMs estimated via Bayesian GMMs may be continuously updated with new data.

Bayesian inference has not been used as extensively as ML based methods to estimate empirical GMMs, possibly because it is seen as more complicated, and people are not familiar with it. Here, I want to bridge that gap, and show how GMMs can be fit using the **R** package R-INLA (https://www.r-inla.org/) (Rue et al. 2017), where INLA stands for *integrated nested Laplace approximation* (Rue, Martino, and Chopin 2009). INLA can be used to fit linear mixed (and many more) models, similar to **lme4**. Most of the aforementioned Bayesian models are estimated via Markov Chain Monte Carlo estimation. By contrast, INLA is a deterministic Bayesian inference approximation approach. INLA is generally quite fast compared to MCMC methods. It is possible to estimate nonergodic GMMs based on varying coefficient models (Landwehr et al. 2016; Bussas et al. 2017; Franco-Villoria, Ventrucci, and Rue 2019), which is quite fast with INLA (Kuehn 2021).

The goal of this tutorial is that an analyst who has fitted models with **lme4** or a similar tool will be able to fit similar models with INLA. The tutorial is not an introduction to Bayesian inference, or to the mathematical background of INLA. This tutorial covers the basic aspects of R-INLA that are needed to estimate GMMs

- Estimate mixed effects models (coefficients, standard deviations, random effects)
- Access results from a fit
- Set prior distributions and control options for the model fit
- Predict new data
- Model comparison/selection

There is a lot more to estimating models with INLA that is not included in this tutorial. For more information on computation with INLA in **R**, see e.g. Gómez-Rubio (2020) (https://becarioprecario.bitbucket.io/inla-gitbook/index.html) or Krainski et al. (2019) (https://becarioprecario.bitbucket.io/spde-gitbook/index.html). For mathematical background, see Rue, Martino, and Chopin (2009) and Rue et al. (2017). The discussion forum at https://groups.google.com/g/r-inla-discussion-group is also a helpful resource. For an introduction to Bayesian inference, see e.g. Kruschke (2015) or Gelman et al. (2013).

2 Getting Started

This tutorial uses the R-INLA package for regression. For installation instructions, see https://www.r-inla.org/download-install. The tutorial requires the following **R** packages.

```
# load required packages
library(lme4)
library(ggplot2)
library(INLA)
```

First, we simulate some data. Here, we assume we have 50 events and 20 stations. Each event is recorded at all 20 stations. Events are randomly assigned a magnitude between 4 and 8, and an event term, sampled from a normal distribution with mean zero and standard deviation τ. Similarly, the V_{S30} and station terms are randomly sampled for each station. Then, for each event/station pair, a distance is sampled, and the

median PSA is calculated according to

$$y = c_1 + c_2 M + c_3 (8 - M)^2 + (c_4 + c_5 M) \ln \left[R_{RUP} + h \right] + c_6 R_{RUP} + c_7 \ln \frac{V_{S30}}{400} + \delta B + \delta S + \delta W S$$

```
## simulate data
tau <- 0.5;
phiSS <- 0.5;
phiS2S <- 0.4;

neq <- 50;
nstat <- 20;

### determine M, event term and observed magnitude
dataM <- matrix(nrow = neq,ncol=2);
set.seed(5618);
for(i in 1:neq) {
  mag <- round(runif(1,4,8),2);
  eta <- rnorm(1,0,tau);
  dataM[i,] <- c(mag,eta);
}

### determine VS, station term and observed VS
dataV <- matrix(nrow = nstat,ncol=2);
set.seed(8472);
for(i in 1:nstat) {
  vs <- round(runif(1,log(300),log(1000)),2);
  lambda <- rnorm(1,0,phiS2S);
  dataV[i,] <- c(vs - log(400),lambda);
}

nrec <- nstat;
data <- matrix(nrow=neq * nrec,ncol = 7);
data_x <- matrix(nrow = neq * nrec, ncol = 7)
data_y <- vector(length = neq * nrec)


h <- 6
coeffs <- c(10.925, -0.985, -0.245, -3.245, 0.32, -0.008, -0.5)

set.seed(98765);
k <- 1
for(i in 1:neq) {
  idx <- 1:nstat;
  mag <- dataM[i,1];
  eqt <- dataM[i,2];
  for(j in 1:nrec) {
    dist <- round(runif(1,1,200),2);
    epsilon <- rnorm(1,0,phiSS);

    vs <- dataV[idx[j],1];

    disteff <- dist + h;
    data_x[k,] <- c(1, mag, (8 - mag)^2, log(disteff), mag * log(disteff), dist, vs);
    pga <- coeffs %*% data_x[k,]
```

```
    pga2 <- pga + epsilon + eqt + dataV[idx[j],2];
    data_y[k] <- pga2;
    data[k,] <- c(mag,dist,vs,pga,pga2,i,idx[j]);
    k <- k+1;
  }
}
```

3 Fitting the Model with lmer

First, we fit a linear model using `lmer`, from the **R** package **lme4** (Bates et al. 2015), to the data. The package **lme4** is the successor to the package **nlme** and together these packages have been used quite extensively for the purposes of calibrating ground-motion models in the past. These packages use more traditional maximum-likelihood based techniques with efficient numerical strategies to fit models. To make our GMM linear, we have to fix the parameter `h` (often referred to as pseudo-depth of near-fault-saturation-term). In this example, we fix it to `h = 6`, which is the value used to generate the data.

```
eqid = data[,6]
statid = data[,7]
M <- data_x[,2]
M2 <- data_x[,3]
lnR <- data_x[,4]
MlnR <- data_x[,5]
R <- data_x[,6]
lnVS <- data_x[,7]
Y <- data_y

fit_lmer <- lmer(Y ~ 1 + M + M2 + lnR + MlnR + R + lnVS + (1 | eqid) + (1 | statid))
coeffs_lmer <- fixef(fit_lmer)
summary(fit_lmer)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: Y ~ 1 + M + M2 + lnR + MlnR + R + lnVS + (1 | eqid) + (1 | statid)
##
## REML criterion at convergence: 1714.6
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.3116 -0.5857  0.0251  0.6254  2.9282
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  eqid     (Intercept) 0.16305  0.4038
##  statid   (Intercept) 0.09118  0.3020
##  Residual             0.26218  0.5120
## Number of obs: 1000, groups:  eqid, 50; statid, 20
##
## Fixed effects:
##               Estimate Std. Error t value
## (Intercept) 11.1938460  1.7117234   6.540
## M           -0.9959825  0.2375447  -4.193
## M2          -0.2997963  0.0545729  -5.494
## lnR         -3.0961045  0.1505916 -20.560
```

```
## MlnR         0.2926556  0.0213087  13.734
## R           -0.0075606  0.0008187  -9.235
## lnVS        -0.5708701  0.1913181  -2.984
##
## Correlation of Fixed Effects:
##       (Intr) M      M2     lnR    MlnR   R
## M     -0.989
## M2    -0.903  0.886
## lnR   -0.362  0.355 -0.014
## MlnR   0.346 -0.393  0.011 -0.909
## R      0.115 -0.015  0.013 -0.444  0.061
## lnVS  -0.041 -0.002  0.000 -0.006  0.006  0.001
```

We can see that we can recover the input parameters quite well.

4 Fitting the Model with INLA

Next, we fit the same model using INLA. Overall, the call is quite similar to `lmer`. We provide a formula with fixed effects and random effects, where the target variable is on the left hand side of `~`, and the fixed and random effects are on the right hand side. Random effects are included as `f(ID, model)` in the formula, where `ID` is an index (such as event or station index) connecting the observation to the random effect. Many different random effects models are implemented. Since event and station terms are independent, we use the `"iid"` model. Information about a random effects model can be accessed with `inla.doc("iid")` for the "iid" random effect, or `inla.doc("besag")` for a Besag spatial model. Chapter 3 of Gómez-Rubio (2020) contains more information about random effects models in INLA, as well as tables of implemented models. We also have to provide a data frame that includes all the variables in the model.

```
data_regression <- data.frame(M,M2,lnR,MlnR,R,lnVS,Y,eqid,statid)

fit_inla <- inla(Y ~ 1 + M + M2 + lnR + MlnR + R + lnVS +
                   f(eqid, model= "iid") + f(statid, model = "iid"),
                 data = data_regression
                 )
summary(fit_inla)
```

```
##
## Call:
##    c("inla(formula = Y ~ 1 + M + M2 + lnR + MlnR + R + lnVS + f(eqid, ", "
##    model = \"iid\") + f(statid, model = \"iid\"), data = data_regression)"
##    )
## Time used:
##     Pre = 7.53, Running = 0.749, Post = 0.408, Total = 8.69
## Fixed effects:
##               mean    sd 0.025quant 0.5quant 0.975quant   mode kld
## (Intercept) 11.195 1.699      7.849   11.194     14.538 11.193   0
## M           -0.996 0.236     -1.461   -0.996     -0.532 -0.996   0
## M2          -0.300 0.054     -0.406   -0.300     -0.193 -0.300   0
## lnR         -3.096 0.151     -3.392   -3.096     -2.801 -3.096   0
## MlnR         0.293 0.021      0.251    0.293      0.335  0.293   0
## R           -0.008 0.001     -0.009   -0.008     -0.006 -0.008   0
## lnVS        -0.571 0.186     -0.939   -0.571     -0.203 -0.571   0
##
## Random effects:
##   Name     Model
```

```
##     eqid IID model
##    statid IID model
##
## Model hyperparameters:
##                                          mean    sd 0.025quant 0.5quant
## Precision for the Gaussian observations  3.82 0.177       3.48     3.82
## Precision for eqid                       6.48 1.431       4.08     6.34
## Precision for statid                    12.62 4.267       5.85    12.12
##                                         0.975quant  mode
## Precision for the Gaussian observations       4.18  3.81
## Precision for eqid                            9.68  6.08
## Precision for statid                         22.40 11.10
##
## Expected number of effective parameters(stdev): 67.15(0.854)
## Number of equivalent replicates : 14.89
##
## Marginal log-Likelihood:  -910.62
```

The summary of the INLA fit gives an overall overview of the fit. We can access individual elements, such as summaries of the fixed effects or the hyperparameters as well. Note that for INLA works internally with (log) precisions, not standard deviations. Precision is one over variance, so we convert the estimates (note: the mean give by the output is the mean of the posterior distribution of the precision parameter. Calculating 1/sqrt() with the mean is not the same as the mean of the posterior distribution of the standard deviation, since the mean is an expectation. Median and quantiles can be converted. For a quick assessment of the values, it is good enough). The summary output consists of estimates of the mean, standard deviation, and quantiles of the parameters. The list of quantiles can be changed.

```
knitr::kable(fit_inla$summary.fixed, digits=3,
             caption = "Summary of fixed effects from INLA fit")
```

Table 1: Summary of fixed effects from INLA fit

	mean	sd	0.025quant	0.5quant	0.975quant	mode	kld
(Intercept)	11.195	1.699	7.849	11.194	14.538	11.193	0
M	-0.996	0.236	-1.461	-0.996	-0.532	-0.996	0
M2	-0.300	0.054	-0.406	-0.300	-0.193	-0.300	0
lnR	-3.096	0.151	-3.392	-3.096	-2.801	-3.096	0
MlnR	0.293	0.021	0.251	0.293	0.335	0.293	0
R	-0.008	0.001	-0.009	-0.008	-0.006	-0.008	0
lnVS	-0.571	0.186	-0.939	-0.571	-0.203	-0.571	0

```
knitr::kable(fit_inla$summary.hyperpar, digits=3,
             caption = "Summary of hyperparameters from INLA fit")
```

Table 2: Summary of hyperparameters from INLA fit

	mean	sd	0.025quant	0.5quant	0.975quant	mode
Precision for the Gaussian observations	3.821	0.177	3.482	3.817	4.180	3.812
Precision for eqid	6.477	1.431	4.076	6.341	9.679	6.085
Precision for statid	12.616	4.267	5.853	12.125	22.404	11.099

```r
# Calculate standard deviations from precisions
1/sqrt(fit_inla$summary.hyperpar$mean)
```

```
## [1] 0.5115969 0.3929313 0.2815354
```

4.1 Setting Prior Distributions

In the fit above, we have used `inla` with default options. Next, we change some options. We explicitly specify the likelihood as normal with `family="gaussian"` (this is the default, but other likelihoods such as Poisson are possible). We also set prior distributions for the parameters. The default prior for the intercept is a normal distribution with mean and precision equal to zero, while for the other fixed effects it is a normal distribution with mean zero and precision 0.001. The default prior for the log precision in the `iid` model is a log-Gamma distribution with shape parameter 1 and rate parameter 0.00005. The precisions are internally represented on the log-scale (since they need to be positive), so we need to specify the prior for these parameters on the log-scale. Here, we use a log-Gamma distribution for the log-precision, which corresponds to a Gamma distribution for the precision (and thus an inverse Gamma distribution for the variance). We use shape parameter 2 and rate parameter 0.5, which implies a prior mean of 4 for the precision.

The prior distributions for the fixed effects are specified as named lists for the mean and precision. Here, we specify a different prior for the linear R term, and use the same values for the other coefficients. These are just examples, and in a real application it makes sense to specify prior distributions based on the problem at hand. We also tell `inla` to calculate CPO (Pettit 1990) and WAIC (Watanabe 2013), which can be used for model comparison. We also change the quantiles that are computed for the summary. A description of the control options can be accessed with e.g. `?control.fixed`.

```r
# prior distributions
prior_prec_tau    <- list(prec = list(prior = "loggamma", param = c(2, 0.5)))
prior_prec_phiS2S <- list(prec = list(prior = "loggamma", param = c(2, 0.5)))
prior_prec_phiSS  <- list(prec = list(prior = "loggamma", param = c(2, 0.5)))

prior.fixed <- list(mean.intercept = 0, prec.intercept = 0.01,
                    mean = (list(R=-0.01, default=0)),
                    prec = (list(R=10000, default = 0.01)))

form <- Y ~ 1 + M + M2 + lnR + MlnR + R + lnVS +
  f(eqid, model= "iid", hyper = prior_prec_tau) +
  f(statid, model = "iid", hyper = prior_prec_phiS2S)

fit_inla1 <- inla(form,
                  data = data_regression,
                  family="gaussian",
                  control.fixed = prior.fixed,
                  control.family = list(hyper = list(prec = prior_prec_phiSS)),
                  control.compute = list(cpo = TRUE, waic = TRUE),
                  quantiles = c(0.05, 0.5, 0.95)
                  )
summary(fit_inla1)
```

```
##
## Call:
##    c("inla(formula = form, family = \"gaussian\", data = data_regression,
##    ", " quantiles = c(0.05, 0.5, 0.95), control.compute = list(cpo = TRUE,
##    ", " waic = TRUE), control.family = list(hyper = list(prec =
##    prior_prec_phiSS)), ", " control.fixed = prior.fixed)")
```

```
## Time used:
##     Pre = 6.05, Running = 1.63, Post = 0.498, Total = 8.18
## Fixed effects:
##               mean    sd 0.05quant 0.5quant 0.95quant   mode kld
## (Intercept) 10.842 1.738     7.976   10.846    13.694 10.855   0
## M           -0.948 0.241    -1.344   -0.949    -0.551 -0.950   0
## M2          -0.290 0.056    -0.381   -0.290    -0.198 -0.290   0
## lnR         -3.083 0.150    -3.330   -3.083    -2.836 -3.083   0
## MlnR         0.291 0.021     0.256    0.291     0.326  0.291   0
## R           -0.008 0.001    -0.009   -0.008    -0.006 -0.008   0
## lnVS        -0.569 0.226    -0.940   -0.569    -0.197 -0.569   0
##
## Random effects:
##   Name     Model
##     eqid IID model
##    statid IID model
##
## Model hyperparameters:
##                                          mean    sd 0.05quant 0.5quant 0.95quant
## Precision for the Gaussian observations 3.82 0.177      3.54     3.82      4.12
## Precision for eqid                      5.88 1.249      4.03     5.76      8.11
## Precision for statid                    8.20 2.550      4.62     7.89     12.85
##                                          mode
## Precision for the Gaussian observations 3.81
## Precision for eqid                      5.56
## Precision for statid                    7.29
##
## Expected number of effective parameters(stdev): 67.85(0.718)
## Number of equivalent replicates : 14.74
##
## Watanabe-Akaike information criterion (WAIC) ...: 1571.44
## Effective number of parameters .................: 67.22
##
## Marginal log-Likelihood:  -887.66
## CPO and PIT are computed
##
## Posterior marginals for the linear predictor and
##  the fitted values are computed
```

We see that there are some minor changes in the coefficients using the model with explicitly set prior distributions, but nothing major. This depends on the prior distributions (more informative prior distributions will probably lead to larger changes).

There are many different prior distributions implemented in INLA. A list of priors can be accessed with `names(inla.models()$prior)`, and documentation on a prior can be accessed with `inla.doc("priorname")`. For more information on prior distributions in INLA, see Chapter 5 of Gómez-Rubio (2020).

4.2 Accessing Results

`inla` by default calculates the marginal distributions for all parameters. These can be assessed with `fit_inla1$marginals.fixed` for the fixed effects, and `fit_inla1$marginals.hyperpar` for the random effects. The marginal distributions are also calculated as the parameter is internally represented, i.e. for ϕ_{SS}, ϕ_{S2S}, and τ, they are log precisions. They can be transformed with function `inla.tmarginal`. Below, we extract the posterior distribution for the magnitude scaling coefficient, and plot it with the true value.

Similar, we take the marginal distribution for the log precision of the event terms, and transform it to the standard deviation scale. If a parameter is transformed, one should use the internal representation. To get help on functions that can perform operations on the marginal distributions, see `help("inla.marginal")`.

```
ggplot(as.data.frame(fit_inla1$marginals.fixed$M), aes(x,y)) +
  geom_line() + geom_vline(xintercept = coeffs[2], colour = "red") +
  labs(x = "c_2", y = "posterior") +
  theme(
    axis.title = element_text(size = 20),
    axis.text = element_text(size = 14),
    plot.title = element_text(size = 30)
  )


posterior_tau <- as.data.frame(inla.tmarginal(function(x) sqrt(exp(-x)),
                                              fit_inla1$internal.marginals.hyperpar[[2]]))
ggplot() +
  geom_line(posterior_tau, mapping = aes(x,y)) + geom_vline(xintercept = tau, colour = "red") +
  labs(x = "tau", y = "posterior") +
  theme(
    axis.title = element_text(size = 20),
    axis.text = element_text(size = 14),
    plot.title = element_text(size = 30)
  )
```

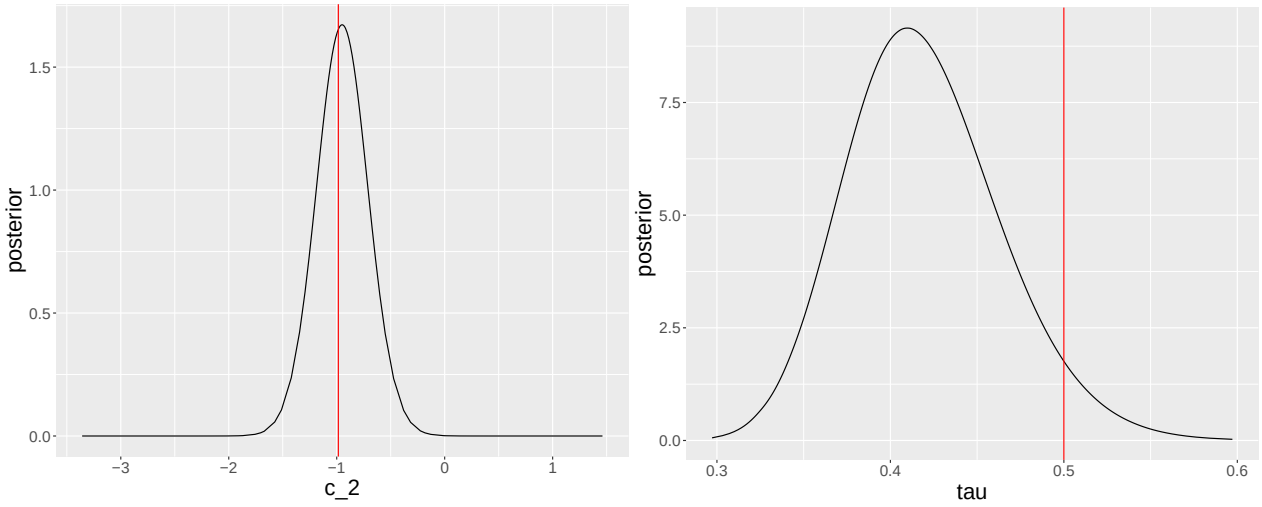

Figure 1: Left: Posterior distribution of coefficient c_2, and true value (red vertical line); Right: Posterior distribution of tau, and true value (red vertical line).

We can also access the random effects for events and stations. They can be similarly accessed from the fitted `inla` object as the fixed effects and hyperparameters. Below, we plot the event terms δB (random effects associated with event id) against magnitude, and against the true values. We also plot uncertainty estimates. By default, the marginal distributions of the random effects are also calculated, and can be accessed via `fit_inla$marginals.random`.

```
deltaB <- fit_inla1$summary.random$eqid

df_plot <- cbind(dataM, deltaB)
```

```r
names(df_plot)[c(1,2)] <- c("M", "deltaB_sim")
names(df_plot)[c(6,7,8)] <- c("q05","q50","q95")

ggplot(df_plot, aes(x = M, y = q50)) +
  geom_point() +
  ylim(-1.2,1.2) +
  geom_pointrange(aes(ymin = q05, ymax = q95)) +
  labs(x = "M", y = "deltaB ") +
  theme(
    axis.title = element_text(size = 20),
    axis.text = element_text(size = 14),
    plot.title = element_text(size = 30)
  )

ggplot(df_plot, aes(x = deltaB_sim, y = q50)) +
  geom_point() + geom_abline(intercept = 0, slope = 1, colour = "red") +
  ylim(-1.2,1.2) +
  geom_pointrange(aes(ymin = q05, ymax = q95)) +
  labs(x = "deltaB_true", y = "deltaB_est") +
  theme(
    axis.title = element_text(size = 20),
    axis.text = element_text(size = 14),
    plot.title = element_text(size = 30)
  )
```

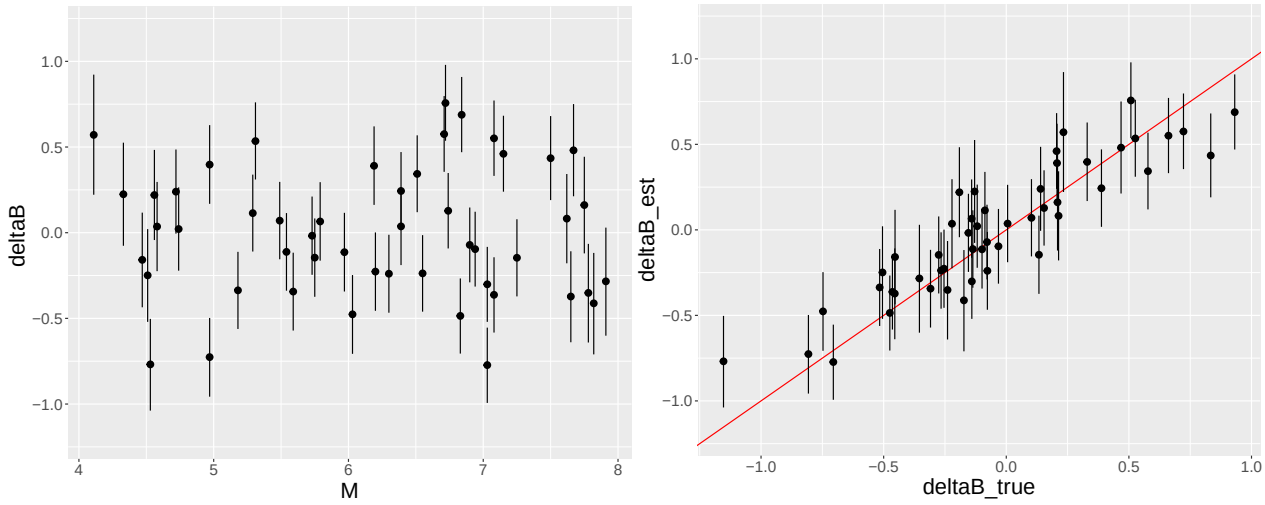

Figure 2: Left: Estimated event terms and associated uncertainty (90% credible interval) against magnitude; Right: Estimated event terms versus simulated event terms.

Fitted values can be accessed with `fit_inla1$summary.fitted.values`, which is useful for calculation of within-event residuals. Here we plot the density of the residuals, but one can just as easily plot them against distance, or another predictor variable.

```r
y_pred <- fit_inla1$summary.fitted.values
resid <- data_y - y_pred$mean

ggplot(as.data.frame(resid), aes(x = resid)) +
```

```
  geom_density() + geom_vline(xintercept = mean(resid), colour = "red") +
  geom_vline(xintercept = sd(resid), colour = "red", linetype="dashed") +
  geom_vline(xintercept = -sd(resid), colour = "red", linetype="dashed") +
  theme(
    axis.title = element_text(size = 20),
    axis.text = element_text(size = 14),
    plot.title = element_text(size = 30)
  )
```

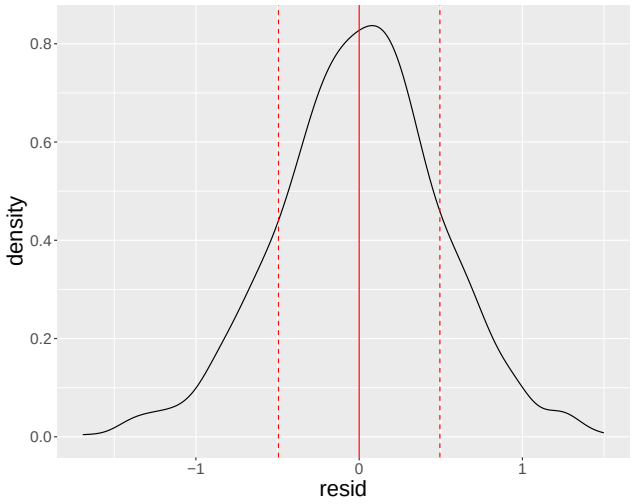

Figure 3: Distribution of within-event residuals.

4.3 Fitting Random Effects with the "Z" Model

So far, we have fitted random effects for event and stations using the `iid` model. Such a model ca also be fit using the `z` model, which requires a matrix of the random effects. In general, the formulation of a random effect is

$$\vec{Y} = \mathbf{X}\vec{\beta} + \mathbf{Z}\vec{u} + \vec{\epsilon}$$

where $\mathbf{X}$ is the design matx, $\vec{\beta}$ is the vector of coeffciets (fixed effects), $\mathbf{Z}$ is the design matrix for the random effects, and $\vec{u}$ are the random effects. For an iid random effect (e.g. event terms), $\mathbf{Z}$ is an $N \times N_{eq}$ matrix, where entry $Z_{ij} = 1$ if the ith records is from the jth event, and zero otherwise. This model is implemented in INLA. It requires the $\mathbf{Z}$ matrix to be passed in the `f()` model. Below, we calculate the $\mathbf{Z}$ marix for event and station terms, and fit the model. We have to set new indices

```
Z_eq <- as(model.matrix(~ 0 + factor(eqid), data = data_regression), "Matrix")
Z_stat <- as(model.matrix(~ 0 + factor(statid), data = data_regression), "Matrix")

data_regression$idx_eq <- 1:length(data_regression$Y)
data_regression$idx_stat <- 1:length(data_regression$Y)

form_z <- Y ~ 1 + M + M2 + lnR + MlnR + R + lnVS +
  f(idx_eq, model= "z", Z = Z_eq, hyper = prior_prec_tau) +
  f(idx_stat, model = "z", Z = Z_stat, hyper = prior_prec_phiS2S)

fit_inla1_z <- inla(form_z,
                    data = data_regression,
                    family="gaussian",
```

```
                control.fixed = prior.fixed,
                control.family = list(hyper = list(prec = prior_prec_phiSS)),
                quantiles = c(0.05, 0.5, 0.95)
                )
```

```
## Warning in inla.model.properties.generic(inla.trim.family(model), mm[names(mm) == : Model 'z' in sec
##   Use this model with extra care!!! Further warnings are disabled.
```

```
summary(fit_inla1_z)
```

```
##
## Call:
##    c("inla(formula = form_z, family = \"gaussian\", data =
##    data_regression, ", " quantiles = c(0.05, 0.5, 0.95), control.family =
##    list(hyper = list(prec = prior_prec_phiSS)), ", " control.fixed =
##    prior.fixed)")
## Time used:
##     Pre = 6.16, Running = 2.79, Post = 0.45, Total = 9.4
## Fixed effects:
##               mean    sd 0.05quant 0.5quant 0.95quant   mode kld
## (Intercept) 10.842 1.739     7.973   10.846    13.696 10.854   0
## M           -0.948 0.241    -1.344   -0.949    -0.550 -0.950   0
## M2          -0.290 0.056    -0.381   -0.290    -0.198 -0.290   0
## lnR         -3.083 0.150    -3.330   -3.083    -2.836 -3.083   0
## MlnR         0.291 0.021     0.256    0.291     0.326  0.291   0
## R           -0.008 0.001    -0.009   -0.008    -0.006 -0.008   0
## lnVS        -0.569 0.226    -0.940   -0.569    -0.197 -0.569   0
##
## Random effects:
##   Name     Model
##     idx_eq Z model
##    idx_stat Z model
##
## Model hyperparameters:
##                                          mean    sd 0.05quant 0.5quant 0.95quant
## Precision for the Gaussian observations 3.82 0.177      3.54     3.82      4.12
## Precision for idx_eq                    5.88 1.252      4.04     5.76      8.12
## Precision for idx_stat                  8.20 2.548      4.62     7.89     12.85
##                                          mode
## Precision for the Gaussian observations 3.81
## Precision for idx_eq                    5.54
## Precision for idx_stat                  7.29
##
## Expected number of effective parameters(stdev): 67.87(0.718)
## Number of equivalent replicates : 14.73
##
## Marginal log-Likelihood:  -887.70
```

For this simple example, the `z` model takes slightly longer than the `iid` model, and gives (almost) identical results (as expected). The `z` model can be used with a more complicated design matrix for the random effects. For example, a cell-specific atenuation model (Dawood and Rodriguez-Marek 2013; Kuehn, Abrahamson, and Walling 2019) can be modeled with the `z` model (Kuehn 2021). In this case, the **Z** matrix is the matrix that contains the fractions of path length in each cell.

4.4 Prediction

One can make predictions of the fitted model by taking a point estimate of the estimated coefficients (such as mean or median) and calculate predictions according to the functional form. We can also use `inla` to make predictions. If the response variable is `NA`, then it is treated as an unknown parameter during the fit, and its posterior distribution is calculated. Below, we create a new data frame with different magnitude values for which we want to make predictions. The `Y` values are assigned `NA`, and we combine the regression data with the prediction data, and fit the model. Now, the posterior distributions can be accessed from the `fitted.values` objects in the `inla` fit. There are summaries available (mean, standard deviation, quantiles of the posterior), which we use below. With `control.predictor = list(compute = TRUE)` we tell INLA to compute marginal posterior distributions for the predictions of each data point, which can be accessed with `$marginals.fitted.values`.

For the values for which we want to calculate predictions, we set the value of the event and station id to a new value; otherwise, the prediction would include the event/station term for the corresponding id. This means that the calculated posterior distribution for the predictions includes uncertainty in the value of the event and station term for the prediction (i.e. the predictive uncertainty includes τ, ϕ_{S2S}, and uncertainty due to uncertainty in coefficients). We can tell INLA to not consider uncertainty due to the random effects by setting the corresponding id to `NA`.

We calculate predictions both ways, and plot the predictions and uncertainty bands below. The second version (random effects set to `NA`) is associated with much lower uncertainty, since it only considers uncertainty in the fixed effects coefficients. As an alternative, we also calculate predictions using the mean coefficients from the previous fit. These are almost identical to the median predictions we plot here.

A quick sidenote on terminology: In the `inla` call below, we calculate the posterior distribution associated with the median ground-motion prediction for a particular scenario. This posterior distribution has its own mean, standard deviation, median, and quantiles, so the term *median prediction* can be ambiguous in this context.

```
# make data frame with scenarios for prediction
# set Y to NA, and event/station ids to a new id
rrup <- 30
data_pred <- data.frame(M = seq(4,8,0.1),
                        M2 = (8 - seq(4,8,0.1))^2,
                        lnR = log(rrup + h),
                        MlnR = seq(4,8,0.1) * log(30 + h),
                        R = rrup,
                        lnVS = log(400/400),
                        Y = NA,
                        eqid = neq+1,
                        statid = nstat+1,
                        idx_eq = NA,
                        idx_stat = NA
)

data_regression2 <- rbind(data_regression, data_pred)
fit_inla1_pred <- inla(form,
                   data = data_regression2,
                   family="gaussian",
                   control.fixed = prior.fixed,
                   control.family = list(hyper = list(prec = prior_prec_phiSS)),
                   control.compute = list(cpo = TRUE, waic = TRUE),
                   quantiles = c(0.05, 0.5, 0.95),
                   control.predictor = list(compute = TRUE)
)
```

```r
#
data_pred_b <- data_pred
data_pred_b$eqid <- NA
data_pred_b$statid <- NA
data_regression2_b <- rbind(data_regression, data_pred_b)
fit_inla1_pred_b <- inla(form,
                  data = data_regression2_b,
                  family="gaussian",
                  control.fixed = prior.fixed,
                  control.family = list(hyper = list(prec = prior_prec_phiSS)),
                  control.compute = list(cpo = TRUE, waic = TRUE),
                  quantiles = c(0.05, 0.5, 0.95),
                  control.predictor = list(compute = TRUE)
)

# access predictions
n_dat <- length(data_regression[,1])
n_pred <- length(data_pred[,1])
pred <- fit_inla1_pred$summary.fitted.values[(n_dat + 1):(n_dat + n_pred),]
pred_b <- fit_inla1_pred_b$summary.fitted.values[(n_dat + 1):(n_dat + n_pred),]

# predict using mean coefficients from first fit
coeffs_inla <- fit_inla1$summary.fixed$mean
data_pred2 <- data.frame(ic = 1,
                         M = seq(4,8,0.1),
                         M2 = (8 - seq(4,8,0.1))^2,
                         lnR = log(rrup + h),
                         MlnR = seq(4,8,0.1) * log(30 + h),
                         R = rrup,
                         lnVS = log(400/400)
)
pred2 <- as.matrix(data_pred2) %*% coeffs_inla

df_plot <- cbind(pred, data_pred, pred2)
names(df_plot)[c(3,4,5)] <- c("q05","q50","q95")

df_plot_b <- cbind(pred_b, data_pred_b)
names(df_plot_b)[c(3,4,5)] <- c("q05","q50","q95")

ggplot() +
  geom_line(df_plot, mapping = aes(x = M, y = q50)) +
  geom_line(df_plot, mapping = aes(x = M, y = q05), linetype="dashed") +
  geom_line(df_plot, mapping = aes(x = M, y = q95), linetype="dashed") +
  geom_line(df_plot_b, mapping = aes(x = M, y = q05), linetype="dotted") +
  geom_line(df_plot_b, mapping = aes(x = M, y = q95), linetype="dotted") +
  geom_line(df_plot, mapping = aes(x = M, y = pred2), color = "red", linetype="dashed") +
  labs(x = "M", y = "y_pred") +
  theme(
    axis.title = element_text(size = 20),
    axis.text = element_text(size = 14),
    plot.title = element_text(size = 30)
  )
```

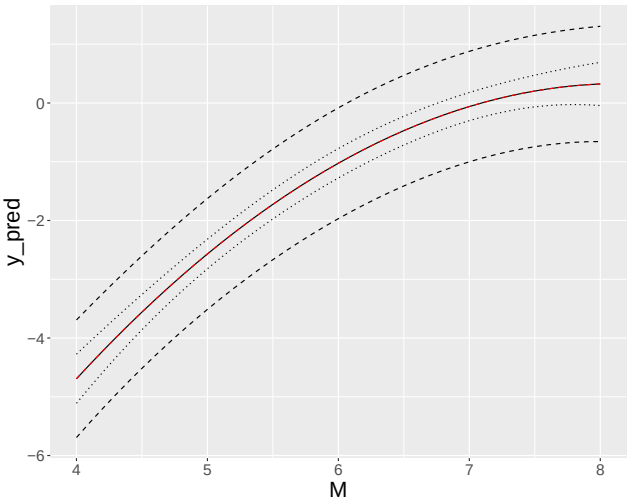

Figure 4: Prediction from the INLA fits. Solid black line shows the median of the posterior distribution associated with predictions. The dashed black lines show the 5%/95% quantiles of the posterior distribution when calculated including random effects uncertainty, while the dotted black lines show the 5%/95% quantiles of the posterior distribution without accounting for random effects. The dotted red line shows th prediction calculated using the mean coefficient values form the previous fit.

4.5 Model Selection

Some model selection criteria are implemented in `inla`. Below, an example is shown using WAIC. We fit a second model, but leave out the magnitude dependent geometrical spreading term. The second model has a much larger WAIC model, which indicates a worse fit.

```
form2 <- Y ~ 1 + M + M2 + lnR + R + lnVS +
  f(eqid, model= "iid", hyper = prior_prec_tau) +
  f(statid, model = "iid", hyper = prior_prec_phiS2S)

fit_inla2 <- inla(form2,
                  data = data_regression,
                  family="gaussian",
                  control.fixed = prior.fixed,
                  control.family = list(hyper = list(prec = prior_prec_phiSS)),
                  control.compute = list(cpo = TRUE, waic = TRUE),
                  quantiles = c(0.05, 0.5, 0.95)
                  )

c(fit_inla1$waic$waic,fit_inla2$waic$waic)
```

```
## [1] 1571.443 1753.249
```

For GMMs, we typically want to make predictions for new events. Hence, one should test predictions if entire events are left out of the data, and do a cross-validation that way. An example is shown below. Here, we partition the events into five folds, and loop over folds. For each iteration, we leave out the data from the test events - this is made easy by replacing their `Y` values with `NA`, and then fit the model. We can then calculate residuals with respect to the estimated predictions on the left out data. In this case, we use the mean estimated predictions.

We also calculate the log-likelihood of the test data. This is more complicated, since we want to average over the posterior distribution. To do that, we sample from the posterior distributions (in this case `n_sample = 1000` times), and calculate the likelihood of the observed data point, given the prediction for each sample. To

calculate the likelihood, we need the prediction, which is stored in `sample[i]]$latent` (for the ith sample), and a value of the standard deviation; the sampled hyperparameters are stored in `sample[[i]]$hyperpar` (in this case, we need the standard deviation of the Gaussian observations, which corresponds to ϕ_{SS}, which is the first hyperparameter). To sample from the joint posterior density, we have to include `control.compute = list(config = TRUE)` in the call to `inla`. The calculation of the log-likelihood in the CV run is taken from a posting by Jonathan Steinhart in the INLA discussion group.

```
library(matrixStats) # for logSumExp

set.seed(1701)
idx_rand <- sample(1:neq)

n_fold <- 5
batch = floor(neq/n_fold)
rest = neq - batch * n_fold
rest_const = rest
n_sample <- 1000

n_mod <- 2
cv_results <- matrix(nrow = n_mod, ncol = n_fold)
lpd_results <- matrix(nrow = n_mod, ncol = n_fold)
resid_mod <- matrix(nrow = n_dat, ncol = n_mod)
loglik_mod <- matrix(nrow = n_dat, ncol = n_mod)

for(b in 1:n_fold) {
  print(paste0('fold ', b))
  if (rest >0){
    batch_t = batch + 1
    beginning = batch_t*(b-1)+1
    end = b*batch_t
    rest = rest - 1
  } else {
    beginning = rest_const + batch*(b-1) + 1
    end = rest_const + batch*b
  }
  idx_test_eq <- idx_rand[beginning:end]
  idx_test <- which(data_regression$eq %in% idx_test_eq)

  y <- data_regression$Y
  y[idx_test] <- NA

  data_reg_p <- data_regression
  data_reg_p$Y <- y

  ### full model
  mod_idx <- 1
  print(paste0('working on model ', mod_idx))
  fit_inla_p <- inla(form,
                     data = data_reg_p,
                     family="gaussian",
                     control.fixed = prior.fixed,
                     control.family = list(hyper = list(prec = prior_prec_phiSS)),
                     control.compute = list(config = TRUE),
                     control.predictor = list(compute = TRUE)
```

```
)

pred <- fit_inla_p$summary.fitted.values[idx_test,]$mean
y_res <- data_regression$Y[idx_test] - pred
cv_results[mod_idx, b] <- sqrt(mean(y_res^2))
resid_mod[idx_test, mod_idx] <- y_res

## sample
sample <- inla.posterior.sample(n_sample, fit_inla_p)
y_sim <- matrix(ncol = length(idx_test), nrow = n_sample)
tau_sim <- vector(length = n_sample)
for(i in 1:n_sample) {
  y_sim[i,] <- sample[[i]]$latent[idx_test]
  tau_sim[i] <- sample[[i]]$hyperpar[1]
}
#subtract true values from each row of y_sim
residuals <- sweep(y_sim, 2, data_regression$Y[idx_test])
#log likelihood of the residual of each draw
log_dens <- dnorm(residuals, sd=sqrt(1/tau_sim), log=TRUE)
#take mean across draws (rows) for each point, giving pointwise likelihoods;
#using log_sum_exp in case of underflow
lpd <- apply(log_dens, 2, function (col) {logSumExp(col) - log(length(col))})
loglik_mod[idx_test, mod_idx] <- lpd
lpd_results[mod_idx, b] <- sum(lpd)

print(paste0('mean = ',mean(y_res), ' sd = ',sd(y_res)))

rm(fit_inla_p)

### model 2
mod_idx <- 2
print(paste0('working on model ', mod_idx))
fit_inla_p <- inla(form2,
                   data = data_reg_p,
                   family="gaussian",
                   control.fixed = prior.fixed,
                   control.family = list(hyper = list(prec = prior_prec_phiSS)),
                   control.compute = list(config = TRUE),
                   control.predictor = list(compute = TRUE)
)

pred <- fit_inla_p$summary.fitted.values[idx_test,]$mean
y_res <- data_regression$Y[idx_test] - pred
cv_results[mod_idx, b] <- sqrt(mean(y_res^2))
resid_mod[idx_test, mod_idx] <- y_res

## sample
sample <- inla.posterior.sample(n_sample, fit_inla_p)
y_sim <- matrix(ncol = length(idx_test), nrow = n_sample)
tau_sim <- vector(length = n_sample)
for(i in 1:n_sample) {
  y_sim[i,] <- sample[[i]]$latent[idx_test]
  tau_sim[i] <- sample[[i]]$hyperpar[1]
```

```
  }

  #subtract true values from each row of y_sim
  residuals <- sweep(y_sim, 2, data_regression$Y[idx_test])
  #log likelihood of the residual of each draw
  log_dens <- dnorm(residuals, sd=sqrt(1/tau_sim), log=TRUE)
  #take mean across draws (rows) for each point, giving pointwise likelihoods;
  #using log_sum_exp in case of underflow
  lpd <- apply(log_dens, 2, function (col) {logSumExp(col) - log(length(col))})
  loglik_mod[idx_test, mod_idx] <- lpd
  lpd_results[mod_idx, b] <- sum(lpd)

  print(paste0('mean = ',mean(y_res), ' sd = ',sd(y_res)))

  rm(fit_inla_p)
}
```

```
## [1] "fold 1"
## [1] "working on model 1"
## [1] "mean = -0.213660238089603 sd = 0.689416132335129"
## [1] "working on model 2"
## [1] "mean = -0.163904289126732 sd = 0.692053107586531"
## [1] "fold 2"
## [1] "working on model 1"
## [1] "mean = -0.118009698384821 sd = 0.627851784743304"
## [1] "working on model 2"
## [1] "mean = -0.124214564484414 sd = 0.692538246173571"
## [1] "fold 3"
## [1] "working on model 1"
## [1] "mean = 0.264159216562142 sd = 0.681641280645511"
## [1] "working on model 2"
## [1] "mean = 0.220407245914732 sd = 0.735219392171675"
## [1] "fold 4"
## [1] "working on model 1"
## [1] "mean = 0.158938302472608 sd = 0.71212035137425"
## [1] "working on model 2"
## [1] "mean = 0.16306271704501 sd = 0.73766638537458"
## [1] "fold 5"
## [1] "working on model 1"
## [1] "mean = -0.039821631605413 sd = 0.574865426359894"
## [1] "working on model 2"
## [1] "mean = -0.0428390063883277 sd = 0.613834887868578"
```

```
row.names(cv_results) <- c("Model 1", "Model 2")
row.names(lpd_results) <- row.names(cv_results)

knitr::kable(cv_results, col.names = c("fold 1", "fold 2", "fold 3", "fold 4", "fold 5"),
             caption = "Root mean square error (RMSE) for individual folds from cross-validation.")
```

Table 3: Root mean square error (RMSE) for individual folds from cross-validation.

	fold 1	fold 2	fold 3	fold 4	fold 5
Model 1	0.7201172	0.6373015	0.7294462	0.7279019	0.5748075

	fold 1	fold 2	fold 3	fold 4	fold 5
Model 2	0.7095121	0.7018835	0.7657834	0.7536713	0.6137951

```
knitr::kable(lpd_results, col.names = c("fold 1", "fold 2", "fold 3", "fold 4", "fold 5"),
             caption = "Log-likelihood for individual folds from cross-validation.")
```

Table 4: Log-likelihood for individual folds from cross-validation.

	fold 1	fold 2	fold 3	fold 4	fold 5
Model 1	-220.4946	-194.1535	-224.1182	-220.4120	-180.2282
Model 2	-216.5561	-212.9191	-231.5036	-228.7375	-192.3830

```
res <- t(rbind(sqrt(colMeans(resid_mod[,]^2)),
                      colSums(loglik_mod)))
row.names(res) <- row.names(cv_results)
knitr::kable(res, col.names = c("RMSE", "LL"),
             caption = "RMSE and total loglikelihood on test set from cross-validation.")
```

Table 5: RMSE and total loglikelihood on test set from cross-validation.

	RMSE	LL
Model 1	0.6807435	-1039.407
Model 2	0.7109482	-1082.099

We can see that the first (the true) model has lower prediction error and higher likelihood on the test set. Hence, both WAIC and cross-validation agree that the first model should be preferred over the second model.

4.6 Partially Nonergodic Model

Here, we show how one one can include multiple random effects in the model, e.g. for regional adjustment terms. We simulate a new data set with 10 regions, where the constant, linear R term, and linear site amplification varies by region.

```
neq <- 25;
nstat <- 20;

nreg <- 10

sigma_c1 <- 0.2
sigma_c6 <- 0.003
sigma_c7 <- 0.3

set.seed(666);
c1_reg <- rnorm(nreg, mean = 0, sd = sigma_c1)
c6_reg <- rnorm(nreg, mean = 0, sd = sigma_c6)
c7_reg <- rnorm(nreg, mean = 0, sd = sigma_c7)

### determine M, event term and observed magnitude
```

```
dataM <- matrix(nrow = neq * nreg,ncol=2);
set.seed(8472);
for(i in 1:(neq * nreg)) {
  mag <- round(runif(1,4,8),2);
  eta <- rnorm(1,0,tau);
  dataM[i,] <- c(mag,eta);
}

### determine VS, station term and observed VS
dataV <- matrix(nrow = nstat * nreg,ncol=2);
set.seed(8472);
for(i in 1:(nstat * nreg)) {
  vs <- round(runif(1,log(300),log(1000)),2);
  lambda <- rnorm(1,0,phiS2S);
  dataV[i,] <- c(vs - log(400),lambda);
}

nrec <- nstat;
data <- matrix(nrow=neq * nrec * nreg,ncol = 8);
data_x <- matrix(nrow = neq * nrec * nreg, ncol = 7)
data_y <- vector(length = neq * nrec * nreg)


h <- 6
coeffs <- c(10.925, -0.985, -0.245, -3.245, 0.32, -0.008, -0.5)

set.seed(98765);
k <- 1
for(r in 1:nreg) {
  coeffs_used <- coeffs
  coeffs_used[1] <- coeffs_used[1] + c1_reg[r]
  coeffs_used[6] <- coeffs_used[6] + c6_reg[r]
  coeffs_used[7] <- coeffs_used[7] + c7_reg[r]
  for(i in 1:neq) {
    idx <- (r - 1) * nstat + 1:nstat;
    mag <- dataM[(r - 1) * neq + i,1];
    eqt <- dataM[(r - 1) * neq + i,2];
    for(j in 1:nrec) {
      dist <- round(runif(1,1,200),2);
      epsilon <- rnorm(1,0,phiSS);

      vs <- dataV[idx[j],1];

      disteff <- dist + h;
      data_x[k,] <- c(1, mag, (8 - mag)^2, log(disteff), mag * log(disteff), dist, vs);
      pga <- coeffs_used %*% data_x[k,]
      pga2 <- pga + epsilon + eqt + dataV[idx[j],2];
      data_y[k] <- pga2;
      data[k,] <- c(mag,dist,vs,pga,pga2,(r - 1) * neq + i,idx[j], r);
      k <- k+1;
    }
  }
}
```

Below is the `inla` call to fit the model. We now have three additional random effects, which are all assumed to be independent, so we use the `"iid"` model for each of them. For the regional terms associated with the anelastic attenuation and the site scaling (scaling with linear R and $\ln V_{S30}$), the covariates are added in the `f()` term after the index. We have to give the region indices a different name, because they occur in multiple `f()` models. We use a PC prior (Simpson et al. 2017) for the precisions of the regional random effects. The PC-prior treats the random effect as a more complex extension of the simpler base model (without regional random effects), and penalizes the more complex model (i.e. a regional random effect should only be estimated if there is a strong signal in the data). The hyperparameters of the PC-prior are two values u and α such that $P(\sigma > u) = \alpha$, which means we set a value of the standard deviation u, and the associated probability α with we believe that σ will be larger than u.

In the fit below, the regional random effects are independent. Correlated random effects can be modeled with `f(id, "iid2d")` (for two random effects) or `f(id, "iid3d")` (for three random effects) models (up to `f(id, "iid5d")`). Information on how to use these random effects models can be obtained via `inla.doc("iid3d")`.

```
eqid = data[,6]
statid = data[,7]
regid = data[,8]
M <- data_x[,2]
M2 <- data_x[,3]
lnR <- data_x[,4]
MlnR <- data_x[,5]
R <- data_x[,6]
lnVS <- data_x[,7]
Y <- data_y

data_regression <- data.frame(M,M2,lnR,MlnR,R,lnVS,Y,eqid,statid,regid,
                              regid_vs = regid, regid_r = regid)

form_reg <- Y ~ 1 + M + M2 + lnR + MlnR + R + lnVS +
  f(eqid, model= "iid", hyper = prior_prec_tau) +
  f(statid, model = "iid", hyper = prior_prec_phiS2S) +
  f(regid, model = "iid",
    hyper = list(prec = list(prior = 'pc.prec', param = c(0.5, 0.01)))) +
  f(regid_vs, lnVS, model = "iid",
    hyper = list(prec = list(prior = 'pc.prec', param = c(0.4, 0.01)))) +
  f(regid_r, R, model = "iid",
    hyper = list(prec = list(prior = 'pc.prec', param = c(0.01, 0.01))))

fit_inla_reg <- inla(form_reg,
                 data = data_regression,
                 family="gaussian",
                 control.fixed = prior.fixed,
                 control.family = list(hyper = list(prec = prior_prec_phiSS)),
                 )

summary(fit_inla_reg)
```

```
##
## Call:
##    c("inla(formula = form_reg, family = \"gaussian\", data =
##    data_regression, ", " control.family = list(hyper = list(prec =
##    prior_prec_phiSS)), ", " control.fixed = prior.fixed)")
## Time used:
##     Pre = 10.5, Running = 14.8, Post = 0.588, Total = 25.8
```

```
## Fixed effects:
##               mean    sd 0.025quant 0.5quant 0.975quant   mode kld
## (Intercept) 11.206 0.855      9.527   11.206     12.883 11.207   0
## M           -1.017 0.118     -1.250   -1.017     -0.785 -1.017   0
## M2          -0.249 0.027     -0.303   -0.249     -0.196 -0.249   0
## lnR         -3.310 0.063     -3.434   -3.310     -3.186 -3.310   0
## MlnR         0.326 0.009      0.308    0.326      0.344  0.326   0
## R           -0.007 0.001     -0.010   -0.007     -0.005 -0.007   0
## lnVS        -0.690 0.107     -0.901   -0.690     -0.478 -0.690   0
##
## Random effects:
##   Name     Model
##     eqid IID model
##    statid IID model
##    regid IID model
##    regid_vs IID model
##    regid_r IID model
##
## Model hyperparameters:
##                                           mean       sd 0.025quant 0.5quant
## Precision for the Gaussian observations    3.86 8.10e-02       3.70     3.86
## Precision for eqid                         4.01 4.10e-01       3.26     3.99
## Precision for statid                       5.57 6.45e-01       4.42     5.53
## Precision for regid                       18.98 1.17e+01       5.42    16.12
## Precision for regid_vs                  1101.34 1.41e+04      11.76   127.27
## Precision for regid_r                  86747.49 3.15e+04   39299.37 82160.99
##                                        0.975quant     mode
## Precision for the Gaussian observations   4.02e+00     3.85
## Precision for eqid                        4.88e+00     3.95
## Precision for statid                      6.95e+00     5.45
## Precision for regid                       4.93e+01    11.75
## Precision for regid_vs                    6.99e+03    24.02
## Precision for regid_r                     1.61e+05 73263.25
##
## Expected number of effective parameters(stdev): 429.74(1.43)
## Number of equivalent replicates : 11.63
##
## Marginal log-Likelihood:  -4458.70
```

5 Real Data Example

Here, we provide a real data example. We use the Italian PGA data that was used in Caramenti et al. (2020) and Lanzano et al. (2021) (https://github.com/lucaramenti/ms-gwr), which is a subset of the data from the ITA18 model Lanzano et al. (2019). Caramenti et al. (2020) estimated a nonergodic model using the Italian, so they discarded some of the global events that were used in Lanzano et al. (2019). The model of Lanzano et al. (2019) has a linear functional form and is estimated with **lme4**, so it provides a good comparison. The

functional form is

$$\begin{aligned}\log_{10} PGA = a &+ b_1(M_W - M_h)\, 1_{(M_w \leq M_h)} + b_2(M_W - M_h)\, 1_{(M_w > M_h)} \\ &+ [c_2 + c_1(M_W - M_h)] \log_{10} \sqrt{R_{JB}^2 + h^2} + c_3 \sqrt{R_{JB}^2 + h^2} \\ &+ k \left[\log_{10} \frac{V_{S30}}{800}\, 1_{(V_{S30} \leq 1500)} + \frac{1500}{800}\, 1_{(V_{S30} > 1500)} \right] + f_1 F_S S + f_2 F_R \\ &+ \delta B + \delta S2S\end{aligned}$$

where $1_{(M_w \leq M_h)}$ is an indicator function that evaluates to one if the condition is true, and zero otherwise. F_{SS} and F_R are indicators for strike-slip and reverse-faulting event, respectively.

First, we read the data set and create the linear covariates.

```
dataset = readRDS(file.path("DATA","italian_data_pga.RData"))

# create linear predictors
mh = 5.5
mref = 5.324
h = 6.924
rm(mag)
attach(dataset)
b1 = (mag-mh)*(mag<=mh)
b2 = (mag-mh)*(mag>mh)
c1 = (mag-mref)*log10(sqrt(JB_complete^2+h^2))
c2 = log10(sqrt(JB_complete^2+h^2))
c3 = sqrt(JB_complete^2+h^2)
f1 = as.numeric(fm_type_code == "SS")
f2 = as.numeric(fm_type_code == "TF")
k = log10(vs30/800)*(vs30<=1500)+log10(1500/800)*(vs30>1500)
y = log10(rotD50_pga)
detach(dataset)

data <- read.csv(file.path("DATA",'italian_data_pga_id.csv'))
n_rec <- length(y)
eq <- data$EQID
stat <- data$STATID
n_eq <- max(eq)
n_stat <- max(stat)


data_italy <- data.frame(Y = y,
                         M1 = b1,
                         M2 = b2,
                         MlnR = c1,
                         lnR = c2,
                         R = c3,
                         Fss = f1,
                         Frv = f2,
                         lnVS = k,
                         eq = eq,
                         stat = stat
)
```

Now, we fit the model using `inla`. We use the same priors and control options as before. We then compare the estimated coefficients with the values from Lanzano et al. (2019). For the hyperparameters (the standard

deviations ϕ_{SS}, τ, and ϕ_{S2S}), we calculate the posterior mean from the transformed marginal distribution using function `inla.emarginal`.

```
form_italy <- Y ~ M1 + M2 + lnR + MlnR + R + Fss + Frv + lnVS +
  f(eq, model = "iid", hyper = prior_prec_tau) +
  f(stat, model = "iid",hyper = prior_prec_phiS2S)

fit_inla_italy <- inla(form_italy,
                  data = data_italy,
                  family="gaussian",
                  control.fixed = prior.fixed,
                  control.family = list(hyper = list(prec = prior_prec_phiSS)),
                  control.compute = list(cpo = TRUE, waic = TRUE),
                  quantiles = c(0.05, 0.5, 0.95)
                  )

# fixed effects with added coefficients from ITA18
coeffs_ita18 <- c(3.4210, 0.1940, -0.0220, -1.4056, 0.2871, -0.0029, 0.0860, 0.0105, -0.3946)
fit_inla_italy$summary.fixed$ITA18 <- coeffs_ita18
knitr::kable(fit_inla_italy$summary.fixed, digits=3,
             caption = "Summary of fixed effects from INLA fit to the Italian data, together with ITA18
```

Table 6: Summary of fixed effects from INLA fit to the Italian data, together with ITA18 coefficients.

	mean	sd	0.05quant	0.5quant	0.95quant	mode	kld	ITA18
(Intercept)	3.412	0.053	3.324	3.412	3.500	3.412	0	3.421
M1	0.205	0.044	0.133	0.205	0.277	0.205	0	0.194
M2	0.001	0.085	-0.139	0.001	0.141	0.001	0	-0.022
lnR	-1.399	0.030	-1.449	-1.399	-1.350	-1.399	0	-1.406
MlnR	0.289	0.014	0.266	0.289	0.312	0.289	0	0.287
R	-0.003	0.000	-0.003	-0.003	-0.003	-0.003	0	-0.003
Fss	0.118	0.043	0.048	0.118	0.188	0.118	0	0.086
Frv	0.001	0.040	-0.064	0.001	0.067	0.001	0	0.011
lnVS	-0.424	0.045	-0.498	-0.424	-0.349	-0.424	0	-0.395

```
# hyperparameters
knitr::kable(fit_inla_italy$summary.hyperpar, digits=3,
             caption = "Summary of hyperparameters from INLA fit to the Italian data.")
```

Table 7: Summary of hyperparameters from INLA fit to the Italian data.

	mean	sd	0.05quant	0.5quant	0.95quant	mode
Precision for the Gaussian observations	24.063	0.554	23.157	24.058	24.984	24.050
Precision for eq	33.734	4.628	26.579	33.471	41.804	33.000
Precision for stat	18.012	1.080	16.286	17.979	19.848	17.912

```
# calculate expected value of standard deviations from posterior marginals of log precisions
tmp <- matrix(c(inla.emarginal(function(x) sqrt(exp(-x)),
                  fit_inla_italy$internal.marginals.hyperpar$`Log precision for the Gaussian observations
  inla.emarginal(function(x) sqrt(exp(-x)),
```

```
                fit_inla_italy$internal.marginals.hyperpar$`Log precision for eq`),
  inla.emarginal(function(x) sqrt(exp(-x)),
                fit_inla_italy$internal.marginals.hyperpar$`Log precision for stat`),
  0.220582, 0.155988, 0.200099), ncol = 3, byrow = TRUE)
row.names(tmp) <- c("INLA","ITA18")
knitr::kable(tmp, col.names = c("phi_SS","tau","phi_S2S"), row.names = TRUE,
             caption = "Estimated standard deviations from INLA fit to the Italian data,
             together with ITA18 values.")
```

Table 8: Estimated standard deviations from INLA fit to the Italian data, together with ITA18 values.

	phi_SS	tau	phi_S2S
INLA	0.2038959	0.1733872	0.2359427
ITA18	0.2205820	0.1559880	0.2000990

The estimated values are similar to the ones from Lanzano et al. (2019), which is not surprising. The standard deviations from INLA are a bit higher, which is probably due to the fact that the number of data is slightly different.

We now plot event and station terms against magnitude and V_{S30}; these are unbiased, as expected.

```
df_plot <- data.frame(M = unique(cbind(eq,dataset$mag))[,2],
                      deltaB = fit_inla_italy$summary.random$eq$mean)
ggplot(df_plot, aes(x = M, y = deltaB)) +
  geom_point() +
  ylim(-0.75, 0.75) +
  #labs(title = "Event Terms") +
  theme(
    axis.title = element_text(size = 20),
    axis.text = element_text(size = 14),
    plot.title = element_text(size = 30)
  )

df_plot <- data.frame(VS = unique(cbind(stat,dataset$vs30))[,2],
                      deltaS = fit_inla_italy$summary.random$stat$mean)
ggplot(df_plot, aes(x = VS, y = deltaS)) +
  geom_point() +
  ylim(-0.75, 0.75) +
  #labs(title = "Station Terms") +
  scale_x_continuous(trans='log10') +
  theme(
    axis.title = element_text(size = 20),
    axis.text = element_text(size = 14),
    plot.title = element_text(size = 30)
  )
```

Lanzano et al. (2019) found that the style-of-faulting coefficients for reverse faulting is not statistically significant at the 0.05 level for PGA. Hence, we fit a second model, but leave out this term, and compare the models via WAIC.

```
form_italy2 <- Y ~ M1 + M2 + lnR + MlnR + R + Fss + lnVS +
  f(eq, model = "iid", hyper = prior_prec_tau) +
  f(stat, model = "iid",hyper = prior_prec_phiS2S)
```

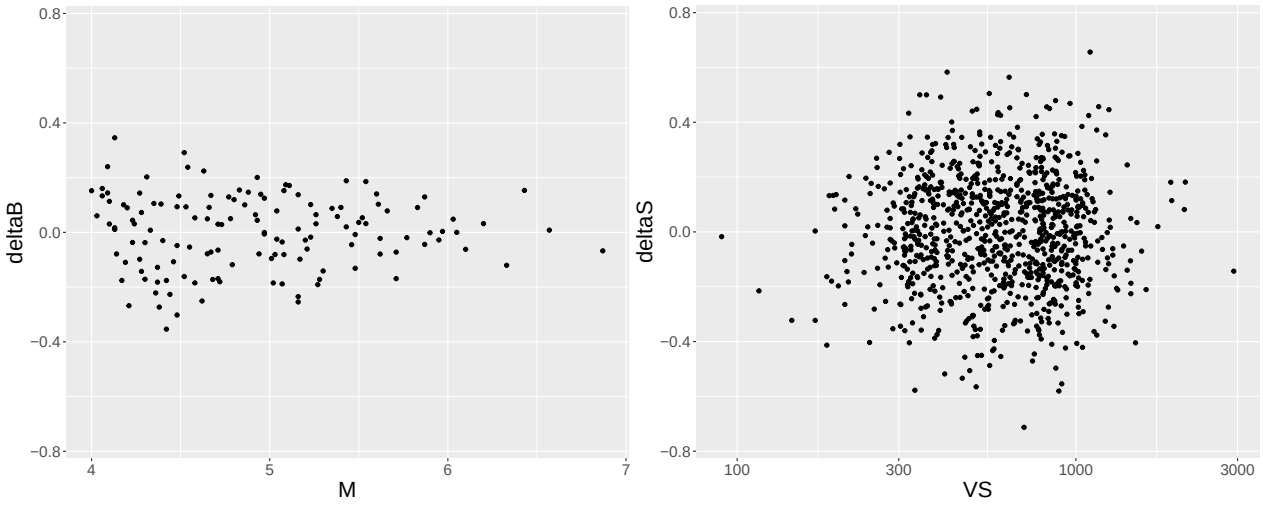

Figure 5: Left: event terms against magnitude for Italian data; Right: station terms agains Vs30 for Italian data.

```
fit_inla_italy2 <- inla(form_italy2,
                  data = data_italy,
                  family="gaussian",
                  control.fixed = prior.fixed,
                  control.family = list(hyper = list(prec = prior_prec_phiSS)),
                  control.compute = list(cpo = TRUE, waic = TRUE),
                  quantiles = c(0.05, 0.5, 0.95)
                  )

c(fit_inla_italy$waic$waic, fit_inla_italy2$waic$waic)
```

```
## [1] -785.0776 -785.9724
```

The two models have almost identical WAIC (the second one has a slightly lower value, meaning it is preferred, thought the difference is small), indicating that the inclusion of f_1 does not improve the model. As indicated by Lanzano et al. (2019), it might still be beneficial to include it, since the effect would be expected to be there (albeit being small), but might not be picked up in the data set at hand.

6 Summary

This tutorial covered the basic steps one has to do to use R-INLA to fit GMMs. This includes estimation of random effects, setting prior distributions, and accessing results. I view INLA as a tool that can be useful in the estimation of GMM; in particular, I believe that the Bayesian paradigm of combining prior information with data has advantageous implications for GMM development. Nevertheless, this is not a philosophical treatise on the relative merits of Bayesian vs. frequentist inference, and as shown in the examples, results obtained by ML estimation using `lmer` can be quite similar to INLA results. This applies both in terms of model coefficients, as well as in model selection, as shown by the Italian example and the style-of-faulting coefficient. In the end, one should use the tool that one has confidence in, it is good to have options. I hope to have piqued interest in INLA, but as stated before, there are a lot more things to INLA than are covered here, and the interested reader is encouraged to seek out some of the references and resources to get a deeper understanding of model estimation with INLA.

One of the reasons to introduce regression with INLA to the GMM development community is that INLA provides fast and efficient methods to estimate spatial models, based on the *stochastic partial differential equations* (SPDE) approach (Lindgren, Rue, and Lindström 2011; Bakka et al. 2018). This allows one to efficiently estimate varying coefficient models (VCMs) (Franco-Villoria, Ventrucci, and Rue 2019). VCMs have been used by Landwehr et al. (2016) to estimate a nonergodic GMM for California, and in Kuehn (2021) I showed how one can estimate a nonergodic GMM based on a VCM efficiently with INLA. Spatial models and (fully) nonergodic GMMs are not covered here, but will be in a future report.

7 Acknowledgements

This tutorial was written using the package **markdown** (Allaire et al. 2019) and rendered with **knitr** (Xie 2014, 2015, 2021). The R Markdown Cookbook (Xie, Dervieux, and Riederer 2020) was a helpful resource for R-Markdown. Conversations with Melanie Walling, Xiaofeng Meng, Grigorios Lavrentiadis, Elnaz Seylabi, Yousef Bozorgnia, Albert Kottke, and Christine Goulet were helpful.

References

Al-Atik, L., N. Abrahamson, J. J. Bommer, F. Scherbaum, F. Cotton, and N. Kuehn. 2010. "The Variability of Ground-Motion Prediction Models and Its Components." *Seismological Research Letters* 81 (5): 794–801. https://doi.org/10.1785/gssrl.81.5.794.

Allaire, J J, Jeffrey Horner, Yihui Xie, Vicent Marti, and Natacha Porte. 2019. *markdown: Render Markdown with the C Library 'Sundown'*. https://cran.r-project.org/package=markdown.

Arroyo, D, and M Ordaz. 2010a. "Multivariate Bayesian Regression Analysis Applied to Ground-Motion Prediction Equations, Part 1: Theory and Synthetic Example." *Bulletin of the Seismological Society of America* 100 (4): 1551–67. https://doi.org/10.1785/0120080354.

———. 2010b. "Multivariate Bayesian Regression Analysis Applied to Ground-Motion Prediction Equations, Part 2: Numerical Example with Actual Data." *Bulletin of the Seismological Society of America* 100 (4): 1568–77. https://doi.org/10.1785/0120090320.

Bahrampouri, Mahdi, Adrian Rodriguez-Marek, and Russell A. Green. 2021. "Ground motion prediction equations for significant duration using the KiK-net database." *Earthquake Spectra* 37 (2): 903–20. https://doi.org/10.1177/8755293020970971.

Bajaj, Ketan, and P. Anbazhagan. 2018. "Determination of GMPE functional form for an active region with limited strong motion data: application to the Himalayan region." *Journal of Seismology* 22 (1): 161–85. https://doi.org/10.1007/s10950-017-9698-5.

Bakka, Haakon, Håvard Rue, Geir-Arne Fuglstad, Andrea Riebler, David Bolin, Janine Illian, Elias Krainski, Daniel Simpson, and Finn Lindgren. 2018. "Spatial modeling with R-INLA: A review." *Wiley Interdisciplinary Reviews: Computational Statistics* 10 (6): e1443. https://doi.org/10.1002/wics.1443.

Bates, Douglas, Martin Maechler, Benjamin M Bolker, and Steven Walker. 2015. "Fitting Linear Mixed-Effects Models using lme4." *Journal of Statistical Software* 67 (1): 1–48. https://doi.org/10.18637/jss.v067.i01.

Bindi, Dino, Fabrice Cotton, Daniele Spallarossa, Matteo Picozzi, and Eleonora Rivalta. 2018. "Temporal Variability of Ground Shaking and Stress Drop in Central Italy: A Hint for Fault Healing?" *Bulletin of the Seismological Society of America* 108 (4): 1853–63. https://doi.org/10.1785/0120180078.

Bindi, Dino, Matteo Picozzi, Daniele Spallarossa, Fabrice Cotton, and Sreeram Reddy Kotha. 2019. "Impact of magnitude selection on aleatory variability associated with ground-motion prediction equations: Part II-analysis of the between-event distribution in central Italy." *Bulletin of the Seismological Society of America* 109 (1): 251–62. https://doi.org/10.1785/0120180239.

Bora, Sanjay Singh, Fabrice Cotton, and Frank Scherbaum. 2019. "NGA-West2 Empirical Fourier and Duration Models to Generate Adjustable Response Spectra." *Earthquake Spectra* 35 (1): 61–93. https://doi.org/10.1193/110317EQS228M.

Bussas, Matthias, Christoph Sawade, Nicolas Kühn, Tobias Scheffer, and Niels Landwehr. 2017. "Varying-coefficient models for geospatial transfer learning." *Machine Learning* 106 (9-10): 1419–40. https://doi.org/10.1007/s10994-017-5639-3.

Campbell, Kenneth W., and Yousef Bozorgnia. 2019. "Ground motion models for the horizontal components of arias intensity (AI) and cumulative absolute velocity (CAV) using the NGA-West2 database." *Earthquake Spectra* 35 (3): 1289–1310. https://doi.org/10.1193/090818EQS212M.

Caramenti, L., A. Menafoglio, S. Sgobba, and G. Lanzano. 2020. "Multi-Source Geographically Weighted Regression for Regionalized Ground-Motion Models." https://mox.polimi.it/publication-results/?id=917%5C&tipo=add_qmox.

Dawood, Haitham M, and Adrian Rodriguez-Marek. 2013. "A Method for Including Path Effects in Ground-Motion Prediction Equations: An Example Using the Mw 9.0 Tohoku Earthquake Aftershocks." *Bulletin of the Seismological Society of America* 103 (2B): 1360–72. https://doi.org/10.1785/0120120125.

Farajpour, Z., S. Pezeshk, and M. Zare. 2019. "A new empirical ground-motion model for Iran." *Bulletin of the Seismological Society of America* 109 (2): 732–44. https://doi.org/10.1785/0120180139.

Franco-Villoria, Maria, Massimo Ventrucci, and Håvard Rue. 2019. "A unified view on Bayesian varying coefficient models." *Electronic Journal of Statistics* 13 (2): 5334–59. https://doi.org/10.1214/19-EJS1653.

Gelman, Andrew, John B Carlin, Hal S Stern, David B Dunson, Aki Vehtari, and Donald B Rubin. 2013. *Bayesian Data Analysis*. February. Chapman; Hall/CRC. https://doi.org/10.1201/b16018.

Gómez-Rubio, Virgilio. 2020. *Bayesian inference with INLA*. Boca Raton, FL.: Chapman; Hall/CRC.

Kotha, Sreeram Reddy, Fabrice Cotton, and Dino Bindi. 2018. "A new approach to site classification: Mixed-effects Ground Motion Prediction Equation with spectral clustering of site amplification functions." *Soil Dynamics and Earthquake Engineering* 110 (July): 318–29. https://doi.org/10.1016/j.soildyn.2018.01.051.

Kotha, Sreeram Reddy, Graeme Weatherill, Dino Bindi, and Fabrice Cotton. 2020. "A regionally-adaptable ground-motion model for shallow crustal earthquakes in Europe." *Bulletin of Earthquake Engineering* 18 (9): 4091–4125. https://doi.org/10.1007/s10518-020-00869-1.

Kowsari, Milad, Benedikt Halldorsson, Birgir Hrafnkelsson, and Sigurjón Jónsson. 2019. "Selection of earthquake ground motion models using the deviance information criterion." *Soil Dynamics and Earthquake Engineering* 117 (May 2018): 288–99. https://doi.org/10.1016/j.soildyn.2018.11.014.

Kowsari, Milad, Tim Sonnemann, Benedikt Halldorsson, Birgir Hrafnkelsson, Jonas P. Snaebjörnsson, and Sigurjon Jonsson. 2020. "Bayesian inference of empirical ground motion models to pseudo-spectral accelerations of south Iceland seismic zone earthquakes based on informative priors." *Soil Dynamics and Earthquake Engineering* 132 (May): 106075. https://doi.org/10.1016/j.soildyn.2020.106075.

Krainski, Elias, Virgilio Gómez-Rubio, Haakon Bakka, Amanda Lenzi, Daniela Castro-Camilo, Daniel Simpson, Finn K. Lindgren, and Håvard Rue. 2019. *Advanced Spatial Modeling with Stochastic Partial Differential Equations Using R and INLA*. Boca-Raton, FL: Chapman; Hall/CRC.

Kruschke, John K. 2015. *Doing Bayesian Data Analysis, Second Edition: A Tutorial with R, JAGS, and Stan*. 2nd Editio. Academic Press / Elsevier.

Kuehn, Nicolas. 2021. "Comparison of Bayesian Varying Coefficient Models for the Development of Nonergodic Ground-Motion Models." *Engrxiv*, 1–25. https://doi.org/10.31224/osf.io/tjxa3.

Kuehn, Nicolas M, and Norman A Abrahamson. 2018. "The Effect of Uncertainty in Predictor Variables on the Estimation of Ground-Motion Prediction Equations." *Bulletin of the Seismological Society of America* 108 (1): 358–70. https://doi.org/10.1785/0120170166.

Kuehn, Nicolas M., and Norman A. Abrahamson. 2020. "Spatial correlations of ground motion for non-ergodic seismic hazard analysis." *Earthquake Engineering & Structural Dynamics* 49 (1): 4–23. https://doi.org/10.1002/eqe.3221.

Kuehn, Nicolas Martin, Norman A Abrahamson, and Melanie Anne Walling. 2019. "Incorporating Nonergodic Path Effects into the NGA-West2 Ground-Motion Prediction Equations." *Bulletin of the Seismological Society of America* 109 (2): 575–85. https://doi.org/10.1785/0120180260.

Kuehn, Nicolas M., and Frank Scherbaum. 2016. "A partially non-ergodic ground-motion prediction equation for Europe and the Middle East." *Bulletin of Earthquake Engineering* 14 (10): 2629–42. https://doi.org/10.1007/s10518-016-9911-x.

Kuehn, N. M., and F. Scherbaum. 2015. "Ground-motion prediction model building: a multilevel approach." *Bulletin of Earthquake Engineering* 13 (9): 2481–91. https://doi.org/10.1007/s10518-015-9732-3.

Landwehr, Niels, Nicolas M. Kuehn, Tobias Scheffer, and Norman Abrahamson. 2016. "A Nonergodic Ground-Motion Model for California with Spatially Varying Coefficients." *Bulletin of the Seismological Society of America* 106 (6): 2574–83. https://doi.org/10.1785/0120160118.

Lanzano, Giovanni, Lucia Luzi, Francesca Pacor, Chiara Felicetta, Rodolfo Puglia, Sara Sgobba, and Maria D'Amico. 2019. "A Revised Ground-Motion Prediction Model for Shallow Crustal Earthquakes in Italy." *Bulletin of the Seismological Society of America* 109 (2): 525–40. https://doi.org/10.1785/0120180210.

Lanzano, Giovanni, Sara Sgobba, Luca Caramenti, and Alessandra Menafoglio. 2021. "Ground-Motion Model for Crustal Events in Italy by Applying the Multisource Geographically Weighted Regression (MS-GWR) Method." *Bulletin of the Seismological Society of America*, August, 1–17. https://doi.org/10.1785/0120210044.

Lindgren, Finn, Håvard Rue, and Johan Lindström. 2011. "An explicit link between gaussian fields and gaussian markov random fields: The stochastic partial differential equation approach." *Journal of the Royal Statistical Society. Series B: Statistical Methodology* 73 (4): 423–98. https://doi.org/10.1111/j.1467-9868.2011.00777.x.

Montalva, Gonzalo A, Nicolás Bastías, and Adrian Rodriguez-Marek. 2017. "Ground-Motion Prediction Equation for the Chilean Subduction Zone." *Bulletin of the Seismological Society of America* 107 (2): 901–11. https://doi.org/10.1785/0120160221.

Ordaz, M, A Arciniega, and S K Singh. 1994. "Bayesian Attenuation Regressions: an Application to Mexico City." *Geophysical Journal International* 117 (2): 335–44. https://doi.org/10.1111/j.1365-246X.1994.tb03936.x.

Parker, Grace A, Jonathan P Stewart, David M Boore, Gail M Atkinson, and Behzad Hassani. 2020. "NGA-Subduction Global Ground-Motion Models with Regional Adjustment Factors." August.

Pettit, L. I. 1990. "The Conditional Predictive Ordinate for the Normal Distribution." *Journal of the Royal Statistical Society: Series B (Methodological)* 52 (1): 175–84. https://doi.org/10.1111/j.2517-6161.1990.tb01780.x.

Rahpeyma, Sahar, Benedikt Halldorsson, Birgir Hrafnkelsson, and Sigurjón Jónsson. 2018. "Bayesian hierarchical model for variations in earthquake peak ground acceleration within small-aperture arrays." *Environmetrics* 29 (3): 1–19. https://doi.org/10.1002/env.2497.

Rue, Håvard, Sara Martino, and Nicolas Chopin. 2009. "Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 71 (2): 319–92. https://doi.org/10.1111/j.1467-9868.2008.00700.x.

Rue, Håvard, Andrea Riebler, Sigrunn H. Sørbye, Janine B. Illian, Daniel P. Simpson, and Finn K. Lindgren. 2017. "Bayesian Computing with INLA: A Review." *Annual Review of Statistics and Its Application* 4 (1): 395–421. https://doi.org/10.1146/annurev-statistics-060116-054045.

Sahakian, Valerie, Annemarie Baltay, Tom Hanks, Janine Buehler, Frank Vernon, Debi Kilb, and Norman Abrahamson. 2018. "Decomposing Leftovers: Event, Path, and Site Residuals for a Small-Magnitude Anza

Region GMPE." *Bulletin of the Seismological Society of America* 108 (5A): 2478–92. https://doi.org/10.1785/0120170376.

Sedaghati, Farhad, and Shahram Pezeshk. 2017. *Bulletin of the Seismological Society of America* 107 (2): 934–48. https://doi.org/10.1785/0120160205.

Simpson, Daniel, Håvard Rue, Andrea Riebler, Thiago G. Martins, and Sigrunn H. Sørbye. 2017. "Penalising Model Component Complexity: A Principled, Practical Approach to Constructing Priors." *Statistical Science* 32 (1): 1–28. https://doi.org/10.1214/16-STS576.

Stafford, Peter J. 2019. "Continuous integration of data into ground-motion models using Bayesian updating." *Journal of Seismology* 23 (1): 39–57. https://doi.org/10.1007/s10950-018-9792-3.

Stafford, P J. 2014. "Crossed and Nested Mixed-Effects Approaches for Enhanced Model Development and Removal of the Ergodic Assumption in Empirical Ground-Motion Models." *Bulletin of the Seismological Society of America* 104 (2): 702–19. https://doi.org/10.1785/0120130145.

Wang, M, and T Takada. 2009. "A Bayesian Framework for Prediction of Seismic Ground Motion." *Bulletin of the Seismological Society of America* 99 (4): 2348–64. https://doi.org/10.1785/0120080017.

Watanabe, Sumio. 2013. "A widely applicable bayesian information criterion." *Journal of Machine Learning Research* 14 (1): 867–97. http://arxiv.org/abs/1208.6338.

Weatherill, Graeme, Sreeram Reddy Kotha, and Fabrice Cotton. 2020. *A regionally-adaptable "scaled backbone" ground motion logic tree for shallow seismicity in Europe: application to the 2020 European seismic hazard model*. Vol. 18. 11. Springer Netherlands. https://doi.org/10.1007/s10518-020-00899-9.

Xie, Yihui. 2014. "knitr: A Comprehensive Tool for Reproducible Research in R." In *Implementing Reproducible Computational Research*, edited by Victoria Stodden, Friedrich Leisch, and Roger D Peng. Chapman; Hall/CRC. http://www.crcpress.com/product/isbn/9781466561595.

———. 2015. *Dynamic Documents with R and knitr*. 2nd ed. Boca Raton, Florida: Chapman; Hall/CRC. https://yihui.org/knitr/.

———. 2021. *knitr: A General-Purpose Package for Dynamic Report Generation in R*. https://yihui.org/knitr/.

Xie, Yihui, Christophe Dervieux, and Emily Riederer. 2020. *R Markdown Cookbook*. 1st Editio. Chapman; Hall/CRC.