Journal of Statistical Software

November 2021, Volume 100, Issue 2. doi: 10.18637/jss.v100.i02

New Frontiers in Bayesian Modeling Using the INLA Package in R

Janet van Niekerk
King Abdullah University
of Science and Technology

Haakon Bakka
King Abdullah University
of Science and Technology

Håvard Rue
King Abdullah University
of Science and Technology

Olaf Schenk
Universita della
Svizzera Italiana

Abstract

The **INLA** package provides a tool for computationally efficient Bayesian modeling and inference for various widely used models, more formally the class of latent Gaussian models. It is a non-sampling based framework which provides approximate results for Bayesian inference, using sparse matrices. The swift uptake of this framework for Bayesian modeling is rooted in the computational efficiency of the approach and catalyzed by the demand presented by the big data era. In this paper, we present new developments within the **INLA** package with the aim to provide a computationally efficient mechanism for the Bayesian inference of relevant challenging situations.

Keywords: **INLA**, joint model, non-separable, spatial, temporal, R.

1. Introduction to the R-INLA project

The R-**INLA** project is an evolving platform that hosts various projects, all interlinked with respect to the **INLA** package (Rue, Martino, and Chopin 2011) in R (R Core Team 2021). This package is based on the INLA methodology developed by Rue, Martino, and Chopin (2009). This development revolutionized the availability and applicability of Bayesian modeling approaches, even in high dimensions, to practitioners and statisticians alike. The INLA methodology ensures computational efficiency by using sparse representations of high dimensional matrices used in latent Gaussian models (LGMs). The computational efficiency of the method offers great appeal to different fields of science and for various applications. In

ecology, Quintero and Jetz (2018) studied bird diversity by using R-**INLA** while Braga, Ter Braak, Thuiller, and Dray (2018) investigated environmental relationships by incorporating phylogenetic information. Dalongeville, Benestan, Mouillot, Lobreaux, and Manel (2018) used R-**INLA** to detect genes specific to salinity in the field of genomics. Air pollution was assessed with the purpose of disease assessment by Shaddick *et al.* (2018) while Rodríguez de Rivera, López-Quílez, and Blangiardo (2018) used R-**INLA** to determine forest species distributions. A study into fire occurrences was conducted by Podschwit, Larkin, Steel, Cullen, and Alvarado (2018) to develop a forecasting system with the use of R-**INLA**. The effect of coral bleaching in the Great Barrier Reef on the marine ecosystem was investigated by Stuart-Smith, Brown, Ceccarelli, and Edgar (2018). In social studies, R-**INLA** has been applied to study the state of education (Graetz *et al.* 2018) and child growth (Osgood-Zimmerman *et al.* 2018) in Africa. These aforementioned works are but a few of many recent applications of R-**INLA**. The pertinency of R-**INLA** is clear. We believe that the new developments presented here will enable more applications in an even broader context.

We present a brief conceptual framework of the INLA methodology. LGMs represent a specific subset of hierarchical Bayesian additive models. This class comprises well-known models such as mixed models, temporal and spatial models. An LGM is defined as a model having a specific hierarchical structure, as follows: The likelihood is conditionally independent based on the likelihood parameters (hyper parameters), $\boldsymbol{\theta}$ and the linear predictors, η_i, such that the complete likelihood can be expressed as

$$\pi(\boldsymbol{Y}|\boldsymbol{\eta},\boldsymbol{\theta}) = \prod_{i=1}^{N} \pi(Y_i|\eta_i(\boldsymbol{\mathcal{X}}),\boldsymbol{\theta}).$$

The linear predictor is formulated as follows:

$$\eta_i = \beta_0 + \boldsymbol{\beta}^\top \boldsymbol{X}_i + \boldsymbol{u}_i(\boldsymbol{z}_i), \tag{1}$$

where $\boldsymbol{\beta}$ represent the linear fixed effects of the covariates $\boldsymbol{X}$ and the unknown non-linear functions $\boldsymbol{u}$ of the covariates $\boldsymbol{z}$ are the structured random effects. These include spatial effects, temporal effects, non-separable spatio-temporal effects, frailties, subject- or group-specific intercepts and slopes, etc. This class of models includes most models used in practice since time series models, spline models and spatial models, amongst others, are all included within this class. The main assumption is that the data, $\boldsymbol{Y}$, is conditionally independent given the partially observed latent field, $\boldsymbol{\mathcal{X}}$, and some hyper parameters $\boldsymbol{\theta}_1$. The latent field $\boldsymbol{\mathcal{X}}$ is formed from the structured predictor as $(\beta_0, \boldsymbol{\beta}, \boldsymbol{u}, \boldsymbol{\eta})$ which forms a Gaussian Markov random field with sparse precision matrix $\boldsymbol{Q}(\boldsymbol{\theta}_2)$, i.e., $\boldsymbol{\mathcal{X}} \sim N(\boldsymbol{0}, \boldsymbol{Q}^{-1}(\boldsymbol{\theta}_2))$. A prior, $\boldsymbol{\pi}(\boldsymbol{\theta})$, can then be formulated for the set of hyper parameters $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)$. The joint posterior distribution is then given by:

$$\boldsymbol{\pi}(\boldsymbol{\mathcal{X}},\boldsymbol{\theta}) \propto \boldsymbol{\pi}(\boldsymbol{\theta})\boldsymbol{\pi}(\boldsymbol{\mathcal{X}}|\boldsymbol{\theta}) \prod_{i=1}^{N} \pi(Y_i|\boldsymbol{\mathcal{X}},\boldsymbol{\theta}). \tag{2}$$

The goal is to approximate the joint posterior density (2) and subsequently compute the marginal posterior densities, $\boldsymbol{\pi}(\mathcal{X}_j|\boldsymbol{Y})$ and $\boldsymbol{\pi}(\boldsymbol{\theta}|\boldsymbol{Y})$. Due to the possibility of a non-Gaussian likelihood, the Laplace approximation is used to approximate this analytically intractable joint posterior density. The sparseness assumption on the precision matrix which characterizes the latent Gaussian field ensures efficient computation (Rue and Held 2005).

In this paper we present some new developments within the **INLA** package in the fields of complex survival models, spatio-temporal models and high performance computing. See Rue *et al.* (2009), Martino, Akerkar, and Rue (2011), Lindgren and Rue (2015), Rue, Riebler, Sørbye, Illian, Simpson, and Lindgren (2017), Bakka *et al.* (2018) and Krainski *et al.* (2019) for more details on some main features of **INLA**. In Section 2 we discuss the implementation of complex survival models including joint longitudinal-survival models, competing risks models and multi-state models. Each of these could incorporate spline, spatial, temporal or clustering elements to mention a few. We then present the new extensions in the spatio-temporal domain, non-separable space-time models in Section 3. Finally, we discuss in Section 4 how the **INLA** package is adapted to a high performance computing environment using the **PARDISO** package (`https://www.pardiso-project.org/`; Alappat *et al.* 2020; Bollhöfer, Schenk, Janalik, Hamm, and Gullapalli 2020; Bollhöfer, Eftekhari, Scheidegger, and Schenk 2019).

2. Complex (joint) survival models

Survival models are used extensively in clinical studies where the time to a certain event is of interest. The hazard function, the instantaneous risk of experiencing the event, is most often of interest to estimate. More importantly, the effects of covariates on the hazard function is of interest for causal inference. Parametric and non-parametric approaches have been proposed to model the hazard function, most are available in the **INLA** package. In this section, we focus on more complex survival models and will not discuss standard survival models (see Martino *et al.* 2011). We present joint longitudinal-survival models in this section, for other complex survival models like competing risk models using **INLA** see Van Niekerk, Bakka, and Rue (2021a).

A basic joint model comprises of two different likelihoods and these likelihoods are joined by shared random effects (see Wulfsohn and Tsiatis 1997; Hu and Sale 2003; Guo and Carlin 2004). Extensions of linear joint models like spatial random effects and non-linear trajectories are used in the context of joint models to address certain practical challenges (see Zhou, Lawson, Hebert, Slate, and Hill 2008; Ratcliffe, Guo, and Ten Have 2004; Andrinopoulou, Eilers, Takkenberg, and Rizopoulos 2018). Each of these new joint models is still an LGM and thus no special implementation package is needed for each one (for more details see Van Niekerk, Bakka, and Rue 2019). Most longitudinal likelihoods and hazard assumptions can be facilitated in this framework, leaving no need to develop a new implementation for each set of assumptions.

2.1. Joint models as LGMs

In this section, we present relevant details of the joint model as an LGM as defined in Section 1, full details are available in Van Niekerk *et al.* (2019). We first present details of the two submodels that form the joint model, a survival and a longitudinal submodel, respectively, and then focus on the joint model in its entirety. Suppose the hazard rate for individual $i, i = 1, \ldots, N^S$ at time s is defined by

$$h_i(s) = h_0(s) \exp(\eta_i^S(s)),$$

where $h_0(s)$ is the baseline hazard function which can be parametrically or non-parametrically

specified and $\eta_i^S(s)$ is the linear predictor (possibly time-dependent), based on covariates, for individual i. Currently, the exponential, Weibull, log-Gaussian and log-Logistic survival distributions are included in the **INLA** package, under the parametric hazard function assumption. The Cox proportional hazards model is included as a semi-parametric model resulting from a non-parametric constant baseline hazard in each of many time partitions (see Cox 1972). In this case, the random walk prior is adopted for the logarithm of the piece-wise constant baseline hazard function, achieving a non-parametric estimate of the baseline hazard function. Now define the density function of the time to event as

$$f_i(s|\eta_i^S(s)) = h_i(s)\exp\left(-\int_0^s h_i(u)du\right),$$

then the likelihood function for the survival submodel is

$$\pi_S(\boldsymbol{s}|\boldsymbol{\eta}^S) = \prod_{i=1}^{N^S}\pi_i(s|\eta_i^S) = \prod_{i=1}^{N^S} f_i(s|\eta_i^S)^{c_i}[1-F_i(s|\eta_i^S)]^{1-c_i}, \tag{3}$$

where $c_i = \mathcal{I}(\text{non-censored observation})$ indicates if an observation is not censored. An observation is censored when the exact event time is not observed but rather the most informative non-event time. Right, left or interval censoring are common censoring approaches and can be accommodated in our approach. The observations are thus a mixture of event times and censored times, depending on the status of each individual.

Now, for the longitudinal data (Y) suppose that each individual has $N_i, i = 1,\ldots,N^S$ observations for a total longitudinal data set size of $N^L = \sum_{i=1}^{N^S} N_i$. We specify the linear predictor $\eta_l^L(t)$, based on covariates at time t, and a conditional density function $g(Y_l|\eta_l^L(t))$ for observation l, resulting in the likelihood for the longitudinal submodel as

$$\pi_L(\boldsymbol{Y}|\boldsymbol{\eta}^L) = \prod_{l=1}^{N^L} g(Y_l|\eta_l^L(t)). \tag{4}$$

Now consider the linear predictors of the joint model,

$$\begin{aligned}\eta_l^{L,J}(t) &= \eta_l^L(t)\\ \eta_i^{S,J}(s) &= \eta_i^S(s) + g(\eta_i^L(s)),\end{aligned} \tag{5}$$

where η^S and η^L are of the form (1) and $g : \Re \to \Re$ is a smooth function of $\eta_i^L(t)$. The function h facilitates the joint estimation of the models and can assume various forms. A common approach is to use the entire longitudinal linear predictor (see Ibrahim, Chu, and Chen 2010), while traditionally only the subject-specific intercept and slope of the time effect have been used, i.e., $g(\eta_i^L(s)) = \nu_1 w_1 + \nu_2 w_2 s$. In the latter we assume the structure specified by Henderson, Diggle, and Dobson (2000) as follows,

$$\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma_{w_1}^2 & \rho\sigma_{w_1}\sigma_{w_2} \\ \rho\sigma_{w_1}\sigma_{w_2} & \sigma_{w_2}^2 \end{bmatrix}\right).$$

Note that either ν_1 or ν_2 can be defined to be zero if desired.

Based on this reconstruction of the joint model, it was demonstrated by Van Niekerk *et al.* (2019) that the joint model is indeed an LGM and can be successfully applied with the **INLA** package.

To this end, we present two illustrative examples. Firstly, we use data from a randomized clinical trial used to investigate the efficacy of two antiretroviral drugs in HIV patients available in the **JMBayes** package (Rizopoulos 2016), where $g(\eta_i^L(s)) = \nu_1 w_1 + \nu_2 w_2 s$. Secondly, we present an example with a non-linear trajectory and informative dropout event process with $g(\eta_i^L(s)) = \nu \eta_i^L(s)$, from a prostate cancer study using post treatment PSA levels as a longitudinal biomarker.

2.2. Example 1: HIV antiretroviral treatments efficacy

In this example the efficacy and safety of two antiretroviral treatments, Didanosine and Zalcitabine, are investigated and presented in Guo and Carlin (2004). This randomized trial includes $N^S = N = 467$ patients who had failed or were intolerant to Zidovudine (AZT) therapy. The longitudinal response is `CD4`, the CD4 cell counts (higher implies healthier), at timepoint `obstime` for each `patient`, such that $N^L = 1405$. Covariates are the `gender`, `prevOI` which denotes if the patient has been diagnosed with AIDS at study entry or not and `AZT` which indicates if there is an intolerance to AZT or if AZT has failed in treatment. The survival responses is the `Time` until death (`death = 1`) or censoring (`death = 0`). In the joint model, we use the same association structure as in Guo and Carlin (2004), i.e.,

$$\begin{aligned} \eta_l^{L,J}(t) &= \eta_l^L(t) + w_1 + w_2 t \\ \eta_i^{S,J}(s) &= \eta_i^S(s) + \nu_1 w_1 + \nu_2 w_2 s. \end{aligned} \tag{6}$$

This model estimates the treatment effect on the survival as well as CD4 count jointly. We can then evaluate the treatments for efficacy in both endpoints by the inclusion thereof as a covariate in both submodels. The specific submodels are then

$$\begin{aligned} \eta_l^{L,J}(t) &= \beta_0^L + \beta_1^L \text{Gender} + \beta_2^L \text{Drug} + \beta_3^L \text{Previous OI} + \beta_4^L \text{AZT Resistance} + w_1 + w_2 t \\ \eta_i^{S,J}(s) &= \beta_0^S + \beta_1^S \text{Gender} + \beta_2^S \text{Drug} + \beta_3^S \text{Previous OI} + \beta_4^S \text{AZT Resistance} + \nu_1 w_1 + \nu_2 w_2 s. \end{aligned}$$

The data is loaded and visualized by the following code (see Figure 1).

```
R> inla.setOption(short.summary = TRUE)
R> data("aids", package = "JMbayes")
R> par(mfrow = c(1, 2))
R> interaction.plot(aids$obstime[1:100], aids$patient[1:100],
+    aids$CD4[1:100], xlab = "Time(years)", ylab = "CD4 count",
+    legend = FALSE, col = 1:100)
R> hist(aids$CD4, main = "", xlab = "CD4 count")
```

In Guo and Carlin (2004) the CD4 counts were transformed with the square root function to use the Gaussian distribution for the response model. In this example we use the original counts and assume a Poisson distribution instead. In Figure 1 it is clear that no zero inflation is evident, although such phenomena could be incorporated into the model using a zero-inflated Poisson distribution for the longitudinal response (available as `zeroinflatedpoisson0` or `zeroinflatedpoisson1` for types 0 and 1, respectively). The individual CD4 trajectories are very different from one another and the need for individual-specific models are clear. This motivates the inclusion of subject-specific intercepts and slopes into the longitudinal

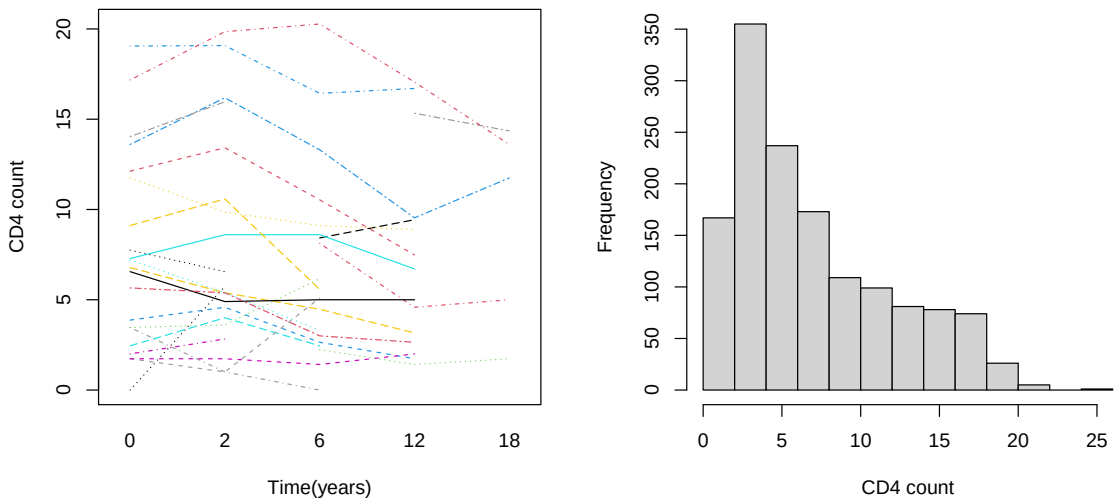

Figure 1: Individual profiles (only first 100 patients) and histogram of CD4 counts.

submodel. In this example we assume the Weibull distribution for the survival times, although the exponential, log-Gaussian, log-Logistic or Cox proportional hazards assumptions could be used as well. We also rescale the time axis to the unit axis using the maximum time for this data set:

```
R> data1 <- aids
R> mtime <- max(data1$Time, data1$obstime)
R> mtime

[1] 21.4

R> data1$Time <- data1$Time / mtime
R> data1$obstime <- data1$obstime / mtime
```

All the times hereafter should thus be rescaled to $(0; 21.4)$ for interpretation.

Preprocessing for joint modeling in ***INLA***

Some preprocessing of the data is required to perform the joint analysis. The full details are omitted here but the concept is illustrated in (7) and the structuring is evident from the forthcoming example code. For each submodel the responses and covariates of the other submodel is stacked with NA's or 0. For the current example define,

$$\boldsymbol{Y} = \begin{bmatrix} y_1 & \text{NA} \\ y_2 & \text{NA} \\ \dots & \dots \\ y_{N^L} & \text{NA} \\ \text{NA} & s_1 \\ \text{NA} & s_2 \\ \dots & \dots \\ \text{NA} & s_N \end{bmatrix}, \qquad \boldsymbol{\beta} = [\beta_1^L \ \beta_1^S \ \dots], \qquad \boldsymbol{X} = \begin{bmatrix} x_{1,1}^L & 0 & \dots \\ x_{1,2}^L & 0 & \dots \\ \dots & 0 & \dots \\ x_{1,N^L}^L & 0 & \dots \\ 0 & x_{1,1}^S & \dots \\ 0 & x_{1,2}^S & \dots \\ 0 & \dots & \dots \\ 0 & x_{1,N}^S & \dots \end{bmatrix},$$

$$\boldsymbol{u}(\boldsymbol{t}) = \begin{bmatrix} w_{1,1} & w_{2,1}t_1 & \text{NA} & \text{NA} \\ w_{1,1} & w_{2,1}t_2 & \text{NA} & \text{NA} \\ \dots & \dots & \dots & \dots \\ w_{1,N} & w_{2,N}t_{n^L} & \text{NA} & \text{NA} \\ \text{NA} & \text{NA} & \nu_1 w_{1,1} & \nu_2 w_{2,1} s_1 \\ \text{NA} & \text{NA} & \nu_1 w_{1,2} & \nu_2 w_{2,2} s_2 \\ \dots & \dots & \dots & \dots \\ \text{NA} & \text{NA} & \nu_1 w_{1,N} & \nu_2 w_{2,N} s_N \end{bmatrix}. \tag{7}$$

Then the joint model in (6) is an LGM similar to (1).

For this example, the construction of the responses and covariates for the joint model is done in the following manner.

First we create the joint fixed effects `fixed.covariate` by padding the covariates with NA's or 0's, for factors or numeric covariates, respectively. The covariates for the joint random effects `random.covariate` are constructed as a list of the padded random covariates. Here all covariates are padded with NA. Since we use model (6), `random.covariate` contains the patient identifiers `patient` for the random intercept and slope effects.

Finally, we construct a list of the longitudinal responses `y.long` and the survival repsonses `y.surv`, which forms our joint response object, `Yjoint`.

Then, the new dataset `jointdataCD4` is constructed from column binding `fixed.covariate` and `random.covariate`, and defining the response `Y` to be the list we created `Yjoint`.

```
R> datas <- data1[data1$obstime == 0, ]
R> datal <- data1[, c(1, 4:12)]
R> ns <- nrow(datas)
R> nl <- nrow(datal)
R> N <- length(unique(data1$patient))
R> fixed.covariate <- data.frame(
+    beta0 = as.factor(c(rep(1, nl), rep(2, ns))),
+    l.drug = as.factor(c(as.factor(datal$drug), rep(NA, ns))),
+    l.gender = as.factor(c(as.factor(datal$gender), rep(NA, ns))),
+    l.prevOI = as.factor(c(as.factor(datal$prevOI) , rep(NA, ns))),
+    l.AZT = as.factor(c(as.factor(datal$AZT) , rep(NA, ns))),
+    s.drug = as.factor(c( rep(NA, nl), as.factor(datas$drug))),
+    s.gender = as.factor(c(rep(NA, nl), as.factor(datas$gender))),
+    s.prevOI = as.factor(c(rep(NA, nl), as.factor(datas$prevOI))),
+    s.AZT = as.factor(c(rep(NA, nl), as.factor(datas$AZT))),
+    l.time = c(datal$obstime, rep(NA, ns)),
+    s.time = c(rep(NA, nl), datas$Time))
R> random.covariate <- list(U11 = c(datal$patient, rep(NA, ns)),
+    U21 = c(datal$patient, rep(NA, ns)),
+    U12 = c(rep(NA, nl), datas$patient),
+    U22 = c(rep(NA, nl), datas$patient))
R> joint.dataCD4 <- c(fixed.covariate,random.covariate)
R> y.long <- c(round(datal$CD4), rep(NA, ns))
R> y.surv <- inla.surv(time = c(rep(NA, nl), datas$Time),
+    event = c(rep(NA, nl), rep(1, ns) - datas$death))
```

```
R> Yjoint <- list(y.long, y.surv)
R> joint.dataCD4$Y <- Yjoint
```

Inference for the joint model

The joint model can be fit using the `inla` function with the defined `formula`. The `family` argument contains the information of the likelihood model(s) and subsequently the appropriate link function(s) for the linear predictor. Since the joint model consists of two likelihoods and hence two linear predictors, we specify the `poisson` distribution for the longitudinal series and the Weibull (`weibullsurv`) distribution for the time to death (in the context of survival data the Weibull distribution is defined by `weibullsurv`). We use the new PC prior for the shape parameter of the Weibull model (Van Niekerk, Bakka, and Rue 2021b), since we do not have any strong prior information.

```
R> JointmodelCD4 = inla(Y ~ -1 + beta0 + l.gender + l.drug + l.prevOI +
+    l.AZT + s.gender + s.drug + s.prevOI + s.AZT +
+    f(U11, model = "iid2d", n = 2 * length(joint.dataCD4$beta0)) +
+    f(U21, l.time, copy = "U11", fixed = TRUE) +
+    f(U12, copy = "U11", fixed = FALSE) +
+    f(U22, s.time, copy = "U11", fixed = FALSE),
+    family = c("poisson", "weibullsurv"), data = joint.dataCD4,
+    verbose = FALSE, control.compute = list(dic = TRUE),
+    control.family = list(list(), list(hyper = list(theta =
+      list(prior = "pc.alphaw", param = c(1))), variant = 1)))
R> summary(JointmodelCD4)

Fixed effects:
            mean    sd 0.025quant 0.5quant 0.975quant   mode kld
beta01     2.310 0.098      2.118    2.309      2.505  2.308   0
beta02     0.279 0.032      0.213    0.280      0.339  0.281   0
l.gender2 -0.007 0.092     -0.186   -0.007      0.174 -0.007   0
l.drug2    0.057 0.054     -0.048    0.058      0.163  0.058   0
l.prevOI2 -0.706 0.067     -0.838   -0.706     -0.574 -0.705   0
l.AZT2    -0.033 0.070     -0.170   -0.033      0.104 -0.033   0
s.gender2 -0.031 0.031     -0.090   -0.032      0.033 -0.034   0
s.drug2   -0.009 0.018     -0.045   -0.009      0.027 -0.009   0
s.prevOI2  0.033 0.022     -0.011    0.034      0.077  0.034   0
s.AZT2    -0.059 0.025     -0.108   -0.059     -0.010 -0.059   0

Model hyperparameters:
                                    mean    sd 0.025quant 0.5quant
alpha parameter for weibullsurv[2]  6.864 0.332      6.269    6.841
Precision for U11 (component 1)     4.207 0.434      3.357    4.212
Precision for U11 (component 2)     0.338 0.124      0.162    0.316
Rho1:2 for U11                     -0.289 0.250     -0.735   -0.297
Beta for U12                       -0.211 0.198     -0.617   -0.204
Beta for U22                        0.332 0.237     -0.112    0.323
```

```
                                  0.975quant   mode
alpha parameter for weibullsurv[2]      7.573  6.777
Precision for U11 (component 1)         5.052  4.257
Precision for U11 (component 2)         0.642  0.278
Rho1:2 for U11                          0.216 -0.307
Beta for U12                            0.159 -0.177
Beta for U22                            0.818  0.290

Deviance Information Criterion (DIC) ...............: 6276.81
Deviance Information Criterion (DIC, saturated) ....: 2053.32
Effective number of parameters .....................: 378.23
```

Similarly to Guo and Carlin (2004), from the estimated joint model we can see that the status of previous AIDS infection (`prevOI`) is a significant covariate in the longitudinal model but not the survival model, and the reason for inclusion in the study (`AZT` failure or intolerance) is a significant covariate in the survival model. Both intercepts are significant. The random intercepts and slopes (w_1 and w_2) can be viewed through their estimated covariance matrix components, $\widehat{\sigma_{w_1}^{-2}}$ = `Precision for U11 (component 1)` = `4.176`, $\widehat{\sigma_{w_2}^{-2}}$= `Precision for U11 (component 2)` = `3.930` and a non-significant correlation of `Rho1:2 for U11` = `0.030`.

The association between the longitudinal and survival models can be investigated by the random values and credible intervals of ν_1 and ν_2 from the summary of the hyper parameters. Both the association parameters are significant, $\widehat{\nu_1}$ = `Beta for U12` = `-0.188` and $\widehat{\nu_2}$ = `Beta for U22` = `0.302`, with respective credible intervals (`-0.342`; `-0.023`) and (`0.061`; `0.531`). The random intercepts are negatively associated and the random slopes are positively associated. A higher starting value of `CD4` thus has a negative effect on the hazard of death, as expected. The positive association of the slopes is also expected since the hazard of death biologically increases over time, as affirmed by the shape parameter estimate of the Weibull model `alpha parameter for weibullsurv[2] = 6.679`. A competing risks model to accomodate death from other causes, might provide more insight.

Patient-specific predictions

To use the model for patient-specific predictions we extract the necessary components from the latent field of the longitudinal and survival submodels. We use the data in `dataH` to calculate the survival functions (corresponding to indices `(nl+1):(nl+ns)`) and `dataL1` to illustrate the observed and estimated longitudinal trajectories (corresponding to indices `1:nl`). The fitted values are extracted using `JointmodelCD4$summary.fitted.values$mean`.

First, the observed event times are extracted as well as the observed longitudinal responses and covariates:

```
R> datas <- data1[data1$obstime == 0, ]
R> datal <- data1[, c(1, 4:12)]
```

Data frames are then created for the estimated survival functions using the observed event times and for the estimated longitudinal trajectories using the observed longitudinal responses and covariates.

```
R> dataH <- data.frame(datas,
+    lambda1 = JointmodelCD4$summary.fitted.values$mean[(nl+1):(nl+ns)])
R> dataL1 <- data.frame(datal,
+    fitted_l = JointmodelCD4$summary.fitted.values$mean[1:nl],
+    random_l = JointmodelCD4$summary.random$U11$mean[1:nl],
+    randoms_l = JointmodelCD4$summary.random$U21$mean[1:nl])
```

For illustration, we calculate the patient-specific CD4 trajectories and survival curves for two patients, one with AIDS at entry and AZT failure (patient 4, `prevOI = 2, AZT = 2`) and one without AIDS at entry and AZT intolerance (patient 35, `prevOI = 1, AZT = 1`), and present these in Figure 2. The solid line is the observed trajectory and the dashed line is the model-based trajectory. We also calculate the parametric model-based survival curve for the Weibull model and indicate the median survival time for each patient with a horizontal line at 0.5.

```
R> patients <- c(4, 35)
R> par(mfrow = c(2, 2))
R> par(mar = c(4, 4, 4, 4))
R> alpha <- JointmodelCD4$summary.hyperpar$mean[1]
R> j_est <- JointmodelCD4$summary.hyperpar$mean[2:6]
R> f_est <- JointmodelCD4$summary.fixed$mean
R> for (patientnr in patients) {
+    dataHi <- dataH[dataH$patient == patientnr, ]
+    with(datal[datal$patient == patientnr, ], plot(obstime * mtime, CD4,
+      ylab = "CD4 count", xlab = "Time (months)", type = "l",
+      xlim = c(0, 21.4), ylim = c(0, 20),
+      main = paste("CD4 trajectory - patient", patientnr)))
+    with(dataL1[dataL1$patient == patientnr, ], lines(obstime * mtime,
+      fitted_l + random_l + randoms_l * obstime, col = "blue", lty = 2))
+    pred_time <- seq(0, 1, by = 0.01)
+    lambda <- dataHi$lambda1
+    plot((pred_time * mtime), exp(-(pred_time * lambda)^alpha),
+      type = "l", ylab = "Survival probability", xlab = "Time (months)",
+      main = paste("Survival curve - patient", patientnr))
+    abline(h = 0.5, col = "red")
+  }
```

2.3. Example 2: PSA levels and informative dropout

Here we use the `prostate` and `dropout` datasets from the **JointModel** package (Kim 2016) for $N^S = 100$ patients with $N^L = 697$ longitudinal observations. The longitudinal response is the logarithm of PSA levels after radiation therapy `logPSA.postRT` at time `VisitTime` for patient with number `ID`. The covariate is the entry PSA level `logPSA.base`. In the `dropout` dataset the time of dropout is given by `DropTime` for patient `ID2` with informative dropout indicator `Status = 1`, and also the entry PSA level `logPSA.base2` as a covariate.

We follow Hu and Sale (2003) and Kim, Zeng, and Taylor (2017) to estimate the longitudinal trajectory by correcting for the bias introduced by the informative dropout. Since the main

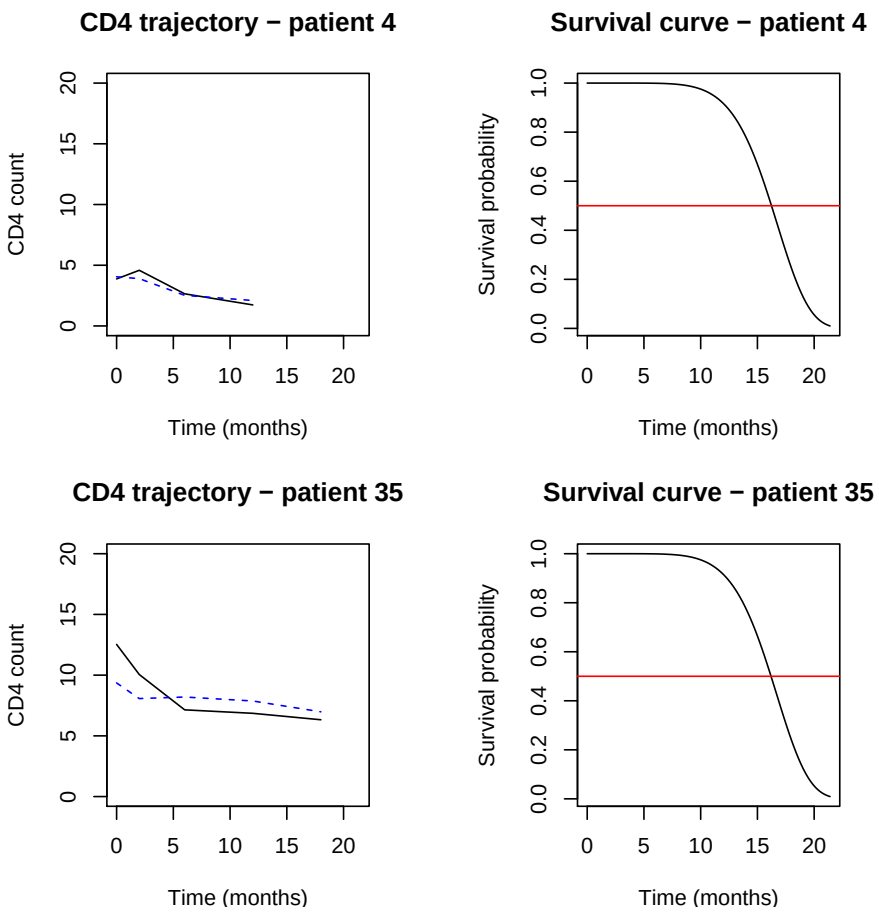

Figure 2: Patient-specific plots – longitudinal trajectories (left, solid line is observed and dashed line is model-based) and model-based survival functions (right).

objective of the analysis is to estimate the non-linear longitudinal trajectories while correcting the bias introduced by informative dropout, we will use the entire longitudinal predictor as the shared random effect, i.e., $h(\eta_l^L(s)) = \nu\eta_l^L(s)$. To model the non-linear trajectory we use a random walk order two component over time $\alpha(t)$. The model is thus:

$$\eta_l^{L,J}(t) = \beta_0^L + \alpha(t) + \beta_1^L PSA_{base}$$
$$\eta_i^{S,J}(s) = \beta_0^S + \nu\eta_i^{L,J}(s),$$

where we assume a Weibull model for the dropout process. Again, we preprocess the original data in the **JointModel** package. The resulting data set is available in the **INLA** package as `exampledata/psa/jointdataPSA.rds`.

Structure of the joint model in ***INLA***

```
R> library("JointModel")
R> inla.setOption(short.summary = TRUE)
R> data1 <- prostate
```

```
R> data2 <- dropout
R> ng <- nrow(data1)
R> ns <- nrow(data2)
R> data1 <- data1[order(data1$VisitTime), ]
R> joint.dataPSA <- readRDS(system.file("exampledata/psa/jointdataPSA.rds",
+    package = "INLA"))
R> joint.dataPSA$beta0 <- joint.dataPSA$mu
R> JointmodelPSA <- inla(Y ~ -1 + beta0 + f(inla.group(V1, n = 50),
+      model = "rw2", scale.model = TRUE,
+      hyper = list(prec = list(prior = "pc.prec", param = c(1, 0.01)))) +
+    b13.PSAbase + f(u, w, model = "iid",
+      hyper = list(prec = list(initial = -6, fixed = TRUE))) +
+    f(b.eta, copy = "u", hyper = list(beta = list(fixed = FALSE))),
+    family = c("gaussian", "gaussian", "weibullsurv"),
+    data = joint.dataPSA, verbose = FALSE,
+    control.compute = list(dic = TRUE, config = TRUE),
+    control.family = list(list(), list(hyper = list(prec =
+      list(initial = 10, fixed = TRUE))), list()))
```

In this case we have three likelihoods. The first set of responses consists of the longitudinal observations, the second of the estimated linear predictors from the longitudinal model and the third of the survival object. This is needed to copy the entire longitudinal linear predictor as a Gaussian random effect (second `"gaussian"`), into the survival submodel (`"weibullsurv"`). The longitudinal linear predictor is added as the second set of responses with a fixed precision of 10 (i.e., variance of 0.1) to ensure that the near-identical linear predictor is copied. The data thus needs to be padded with an additional set of `NA`'s in the second response vector to achieve this specific association term, i.e., $h(\eta_l^L(s)) = \nu\eta_l^L(s)$. Also note the term `f(inla.group(V1, n = 50), model = "rw2")` is the second order random walk two model used as a spline to capture the non-linear effect of time, where we use 50 bins of time to fit this model component. We use the `scale.model = TRUE` option to ensure a generalized variance of 1 (this results in an interpretable precision parameter for different intrinsic Gaussian Markov random fields, like the random walk order two model) and we assume the PC prior for the precision hyperparameter with the option `hyper = list(prec = list(prior = "pc.prec", param = c(1, 0.01)))` (Simpson, Rue, Riebler, Martins, and Sørbye 2017).

Inference for the joint model

We extract the results as follows:

```
R> summary(JointmodelPSA)

Fixed effects:
              mean    sd 0.025quant 0.5quant 0.975quant   mode kld
beta01       0.083 0.038      0.008    0.083      0.158  0.083   0
beta02      -0.985 0.185     -1.364   -0.980     -0.638 -0.968   0
b13.PSAbase  0.421 0.026      0.370    0.421      0.472  0.421   0
```

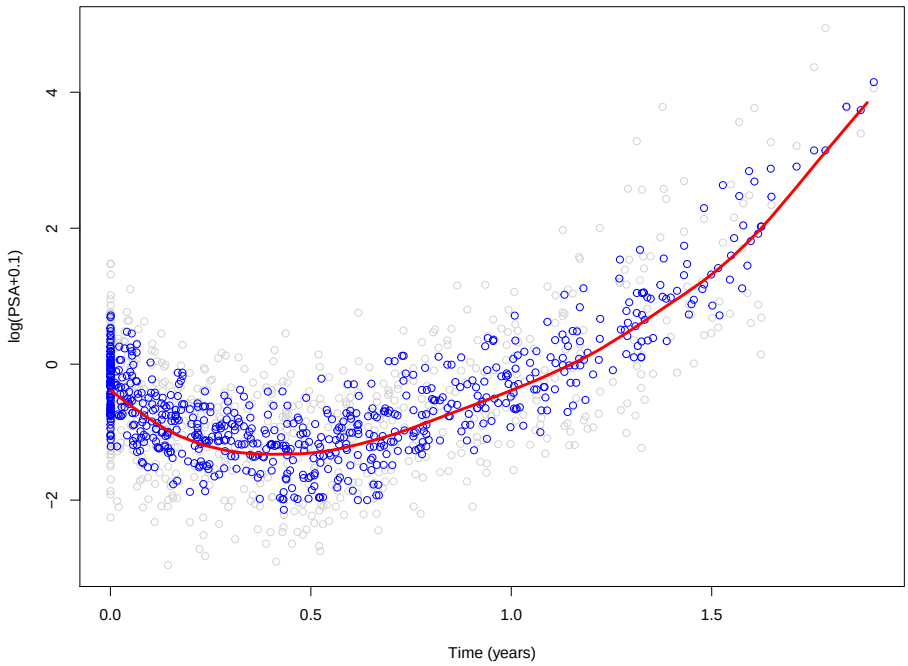

Figure 3: Average PSA trajectory (red line), estimated PSA levels (blue) and observed PSA levels (grey).

```
Model hyperparameters:
                                          mean    sd 0.025quant
Precision for the Gaussian observations 2.084 0.112      1.870
alpha parameter for weibullsurv[3]      0.814 0.068      0.691
Precision for inla.group(V1, n = 50)    5.949 4.325      1.198
Beta for b.eta                          1.166 0.246      0.681
                                        0.5quant 0.975quant  mode
Precision for the Gaussian observations    2.082      2.312 2.079
alpha parameter for weibullsurv[3]         0.809      0.959 0.799
Precision for inla.group(V1, n = 50)       4.847     17.334 3.048
Beta for b.eta                             1.165      1.650 1.165

Deviance Information Criterion (DIC) ...............: -3750.72
Deviance Information Criterion (DIC, saturated) ....: 1186.96
Effective number of parameters .....................: 183.49
```

Both the intercepts and the `logPSA.base` covariate are significant. From the hyper parameters, the spline is significant in its departure from linearity as deduced from the precision of the spline `Precision for inla.group(V1, n = 50)` = `5.643`. The shape parameter for the Weibull model is estimated as `alpha parameter for weibullsurv[3]` = `0.814` which implies a decreasing hazard of dropout over time.

The association parameter $\widehat{\nu}$ = `Beta for b.eta` = `1.167` is significant and it is thus clear that the joint model approach is necessary for this data set, to account for the informative dropout.

For illustration we can extract the spline component with `JointmodelPSA$summary.random$` `inla.group(V1, n = 50)` and the model-based PSA levels with `JointmodelPSA$summary.fitted.values` as in Example 1. The estimated non-linear longitudinal average trajectory is illustrated in Figure 3 by the red line, the model-based PSA levels are given by the blue circles and the grey circles represent the observed PSA levels.

```
R> par(mfrow = c(1, 1))
R> plot(data1$VisitTime, data1$logPSA.postRT, xlab = "Time (years)",
+    ylab = "log(PSA+0.1)", col = "lightgrey")
R> points(data1$VisitTime, JointmodelPSA$summary.fitted.values[1:ng, 1],
+    col = "blue", lwd = 1)
R> lines(JointmodelPSA$summary.random$`inla.group(V1, n = 50)`[1:50, 1],
+    JointmodelPSA$summary.random$`inla.group(V1, n = 50)`[1:50, 2],
+    col = "red", lwd = 3)
```

3. Non-separable space-time models

The **INLA** package has been very successful in space and space-time modeling by representing spatial models with sparse matrices using the stochastic partial differential equations (SPDE) approach (Lindgren, Rue, and Lindström 2011; Bakka *et al.* 2018; Krainski *et al.* 2019). Space-time models are usually constructed as Kronecker products, resulting in separable models, where the space-time covariance function is a product of a spatial and a temporal covariance function. In **INLA** this is coded using the `group` and `control.group` arguments of the function `f`.

Instead of constructing a space-time model as an interaction between a spatial and a temporal model, Bakka, Krainski, Bolin, Rue, and Lindgren (2020) are developing a class of space-time models directly from the principles of diffusion processes in space-time. The basic building block is a Matérn model in space, which is smoothed by a space-time diffusion process. The spatial Matérn model is a natural starting point due to its wide use in spatial modeling in general, and in **INLA** in particular. Define the spatial differential operator

$$L = \left(\gamma_s^2 + \Delta\right),$$

where γ_s^2 is a constant, and $\Delta = (d^2/dx^2, d^2/dy^2)$ is the Laplacian. The space-time diffusion process is governed by the differential operator

$$\left(\gamma_t \frac{d}{dt} + L\right),$$

known as a reaction-diffusion operator in physics, and used in many physical models. This operator is used in systems where mass (which can represent mass, energy, individuals, disease counts, or other characteristics) changes in time due to diffusion and replication.

In this section we discuss an implementation of the new models using the computational methods based on Gaussian Markov random fields (GMRFs), in **INLA**, by writing R code for the inference explicitly instead of using the `inla` function call. The main purpose of this section is to show clearly how the GMRF framework can be used to code inference in the

context of complex spatio-temporal random effects. Further, the reader can study the sparsity structure of the matrices we present to see how well this approach fits with the research on parallel computations presented in Section 4. The code herein is meant for understanding software implementation, and is not suitable for applications, for that we refer to the online code examples in the Supplementary Material of Bakka *et al.* (2020).

```
R> library("fields")
R> library("viridisLite")
R> set.seed(2019)
```

3.1. Define precision matrices

We use simple temporal and spatial meshes as follows. The spatial mesh can be plotted by `plot(mesh)` and is described in Krainski *et al.* (2019).

```
R> t.max <- 8
R> mesh.time <- inla.mesh.1d(1:t.max)
R> fake.locations <- matrix(c(0, 0, 10, 10, 0, 10, 10, 0), nrow = 4,
+    byrow = TRUE)
R> mesh.space <- inla.mesh.2d(loc = fake.locations, max.edge = c(1.5, 2))
```

We use the model DEMF$(1, 2, 1)$ in Bakka *et al.* (2020), corresponding to the SPDE

$$\left(\gamma_t \frac{d}{dt} + L\right) L^{1/2} \gamma_\epsilon u(s,t) = \mathcal{W}(s,t),$$

where the γ's are hyper parameters, and $\mathcal{W}$ is a white noise process.

Before we can define the separable and the non-separable models, we need to decide the hyper parameters for our two space-time models. We choose the following hyper parameters (γ's) to give reasonable random fields, and to make the separable and non-separable models as similar as possible when it comes to scale parameters, see Bakka *et al.* (2020).

We select the following hyper parameters of the random effects for the separable model

```
R> range.time <- 20
R> range.space <- 6
R> sigma.u <- 1
```

and for the non-separable model

```
R> gt <- 2.23
R> gs2 <- 0.2222
R> ge2 <- 0.0805
```

We use the finite element method (FEM) from Lindgren *et al.* (2011), adopted by Bakka *et al.* (2020) to the following M-notation. We note that the temporal model is first order Markov, and that a higher order Markov structure would be used for models with a higher smoothness in time (Bakka *et al.* 2020).

We first preprocess the spatial and temporal FEM matrices.

```
R> sfe <- inla.mesh.fem(mesh.space, order = 4)
R> tfe <- inla.mesh.fem(mesh.time, order = 2)
R> M0 <- tfe$c0
R> N.t <- nrow(M0)
R> M1 <- sparseMatrix(i = c(1, N.t), j = c(1, N.t), x = 0.5)
R> M2 <- tfe$g1
```

Conditional on the chosen hyper parameters, we define the precision matrices (Q) for the separable and the non-separable models.

```
R> kappa <- 2/range.time
R> Q.M <- kappa^2 * M0 + 2 * kappa * M1 + M2
R> Q.M <- Q.M/2/kappa
R> Q.space.alpha2 <- gs2^2 * sfe$c0 + 2 * gs2 * sfe$g1 + sfe$g2
R> Q.space.alpha2 <- Q.space.alpha2/(4 * pi * gs2)
R> Q.separ <- kronecker(Q.M, Q.space.alpha2)
R> Q.nonsep <- (kronecker(gt^2 * M2, gs2 * sfe$c0 + sfe$g1) +
+    kronecker(M0, gs2^3 * sfe$c0 + gs2^2 * sfe$g1 + gs2 * sfe$g2 + sfe$g3) +
+    kronecker(2 * gt * M1, gs2^2 * sfe$c0 + 2 * gs2 * sfe$g1 + sfe$g2 )) *
+    ge2
```

We can study the prior marginal variance as follows. Importantly, we note that the marginal variance is near 1 for both models.

```
R> summary(diag(inla.qinv(Q.separ)))

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.022   1.165   1.311   1.576   1.798   3.141

R> summary(diag(inla.qinv(Q.nonsep)))

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.076   1.255   1.422   1.720   1.990   3.566
```

3.2. Prior simulation

We simulate a Matérn field for $t = 1$. This can be done through the separable or the non-separable model, since they are both Matérn marginally for $t = 1$. We use the `seed` and `num.threads = 1` arguments to get reproducible simulations. Further, we add a small noise to the observations to give a more realistic inference problem. The dataframe `df` is represented for all of space and time, but we replace the observations by `NA` after year 1.

```
R> u <- inla.qsample(n = 1, Q.nonsep, seed = 2019, num.threads = 1)[, 1]
R> N.st <- length(u)
R> sig.eps <- 0.01
R> noise <- rnorm(N.st, 0, 1) * sig.eps
R> df <- data.frame(y = u * sigma.u + noise, st = 1:N.st)
R> df$y[-(1:mesh.space$n)] <- NA
```

Note that `N.st` is equal to `nrow(Q.separ)` and `nrow(Q.nonsep)`.

3.3. Inference in R

We follow the book Rue and Held (2005) and compute posterior precision matrices and means, by conditioning on `df$y`. The following code chunk replaces the call to `inla` in this example, showing how the sparsity of the precision matrices impacts the main computational part of the **INLA** inference procedure.

First the precision matrix for observation noise `Qeps`, the projection matrix for the observed latent field `A.observe` and the posterior/conditional precision matrix `post.Q` are specified.

```
R> Qeps <- Diagonal(n = mesh.space$n)
R> A.observe <- sparseMatrix(i = 1:mesh.space$n, j = 1:mesh.space$n,
+    dims = c(mesh.space$n, N.st))
R> post.Q <- function(sig.eps = 0.01, Q.model) {
+    Q <- sig.eps^{-2} * t(A.observe) %*% Qeps %*% A.observe + Q.model
+    return(Q)
+  }
```

Then the point estimates are determined using the posterior means.

```
R> post.mu <- function (sig.eps = 0.01, Q.model) {
+    a <- df$y[1:mesh.space$n]
+    b <- sig.eps^{-2} * t(A.observe) %*% Qeps %*% a
+    res <- inla.qsolve(post.Q(sig.eps, Q.model = Q.model), b)
+    return(res)
+  }
R> mu.post.separ <- post.mu(Q.model = Q.separ)
R> mu.post.nonsep <- post.mu(Q.model = Q.nonsep)
```

For convenience, we set up a local function for plotting, designed for our example. This is developed from the code in Krainski *et al.* (2019).

```
R> local.plot.field <- function(field, mesh, time = 1, ...) {
+    field <- field[1:mesh$n + (time-1) * mesh$n]
+    proj <- inla.mesh.projector(mesh, dims = c(200, 200))
+    field.proj <- inla.mesh.project(proj, field)
+    image.plot(list(x = proj$x, y = proj$y, z = field.proj),
+      col = plasma(64), ...)
+  }
```

The function first subsets `field` to use only the relevant part of the incoming vector and then projects the mesh onto a 200×200 grid.

We plot the point predictions (posterior mean) in space-time, in Figure 4. Note that in year 1 the field is conditioned on data on nearby locations, hence the separable and the non-separable models give very similar results. Year 2 and 3, however, represent forecasts based on the data observed in year 1. The plots shown here are for the first three years, but the for loop can

be extended to show all six. In the figure we see a clear difference between the separable and the non-separable models. The separable model forecasts a simple decay to the mean of the current observations, while the non-separable model results in smoother forecasts. Due to the very long temporal range, the decay to the mean is hard to spot visually. We argue that the non-separable forecast is more appropriate in most applied situations. When forecasting, e.g., the temperature in a location in the future, the model should use not just the temperature in the same location today, but also use the temperature in nearby locations, resulting in a smoother forecast. One classical example of this is hot water poured into cold water; we expect the two temperatures to regress to the mean by mixing and smoothing out differences.

```
R> par(mfrow = c(3, 2))
R> zlim2 <- range(c(mu.post.separ, mu.post.nonsep))
R> for (tp in 1:3) {
+    local.plot.field(mu.post.separ, mesh.space, time = tp,
+      main = paste0("Separable mean, t=", tp),
+      xlim = c(0, 10), ylim = c(0, 10), zlim = zlim2)
+    local.plot.field(mu.post.nonsep, mesh.space, time = tp,
+      main = paste0("Non-separable mean, t=", tp),
+      xlim = c(0, 10), ylim = c(0, 10), zlim = zlim2)
+  }
```

3.4. Posterior simulations

Finally, we show how to simulate from the posterior, in Figure 5. As before, the first year is very similar, because we conditioned on data here, while year 2 and 3 show different simulations into the future. This code is a replacement for `inla.posterior.sample`, showing the similarity between posterior simulations for the separable and non-separable models.

```
R> post.sim.separ <- inla.qsample(1, Q = post.Q(Q.model = Q.separ),
+    reordering = "identity", seed = 1, num.threads = 1)
R> post.sim.separ = drop(post.sim.separ + mu.post.separ)
R> post.sim.nonsep = inla.qsample(1, Q = post.Q(Q.model = Q.nonsep),
+    reordering = "identity", seed = 1, num.threads = 1)
R> post.sim.nonsep <- drop(post.sim.nonsep + mu.post.nonsep)
```

Then we plot the simulations.

```
R> zlim1 <- range(c(post.sim.separ, post.sim.nonsep))
R> par(mfrow = c(3, 2))
R> for (tp in c(1, 2, 3)) {
+    local.plot.field(post.sim.separ, mesh.space, time = tp,
+      main = paste0("Separable sim, t=", tp),
+      xlim = c(0, 10), ylim = c(0, 10), zlim = zlim1)
+    local.plot.field(post.sim.nonsep, mesh.space, time = tp,
+      main = paste0("Non-separable sim, t=", tp),
+      xlim = c(0, 10), ylim = c(0, 10), zlim = zlim1)
+  }
```

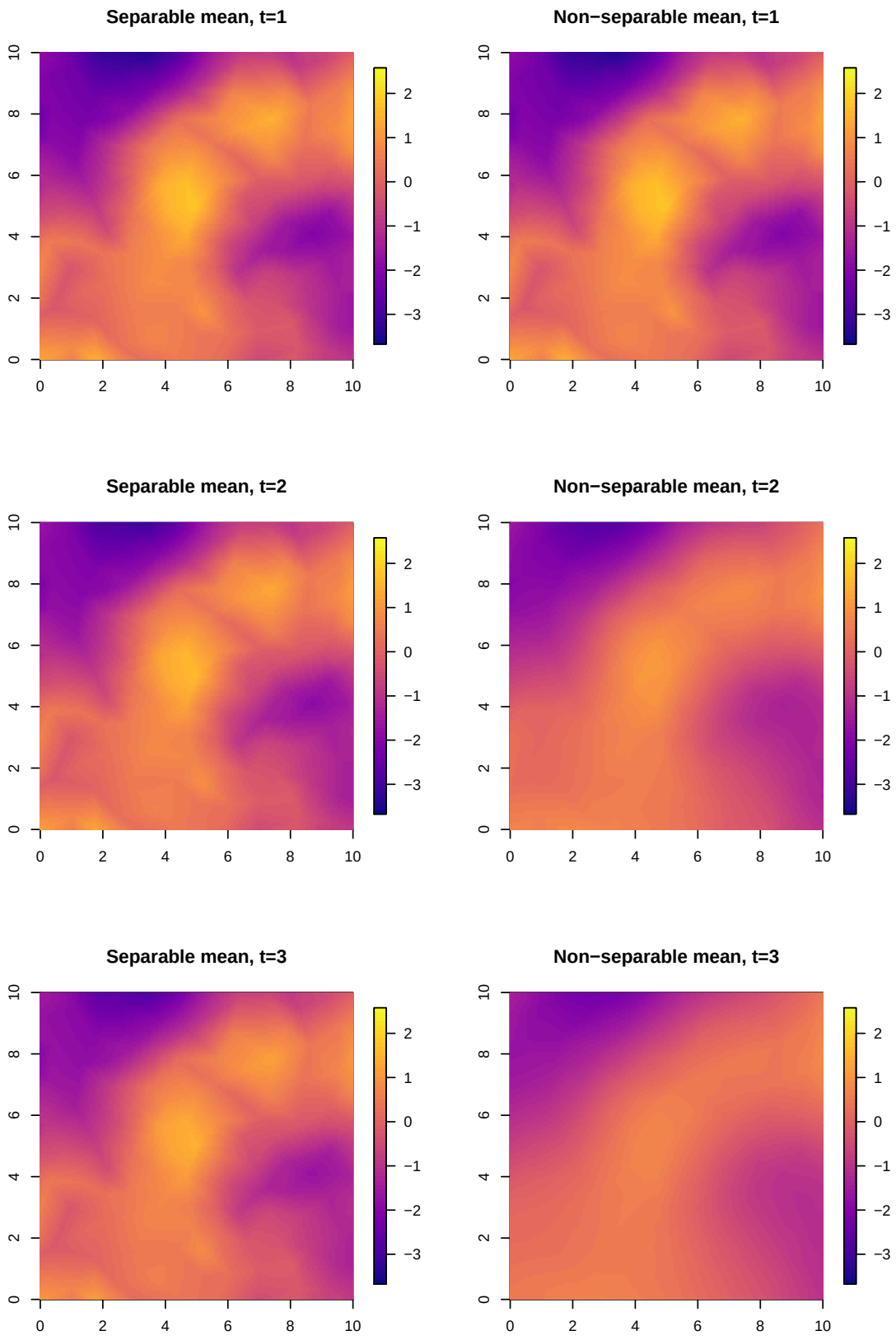

Figure 4: Posterior mean estimates from the separable and non-separable models.

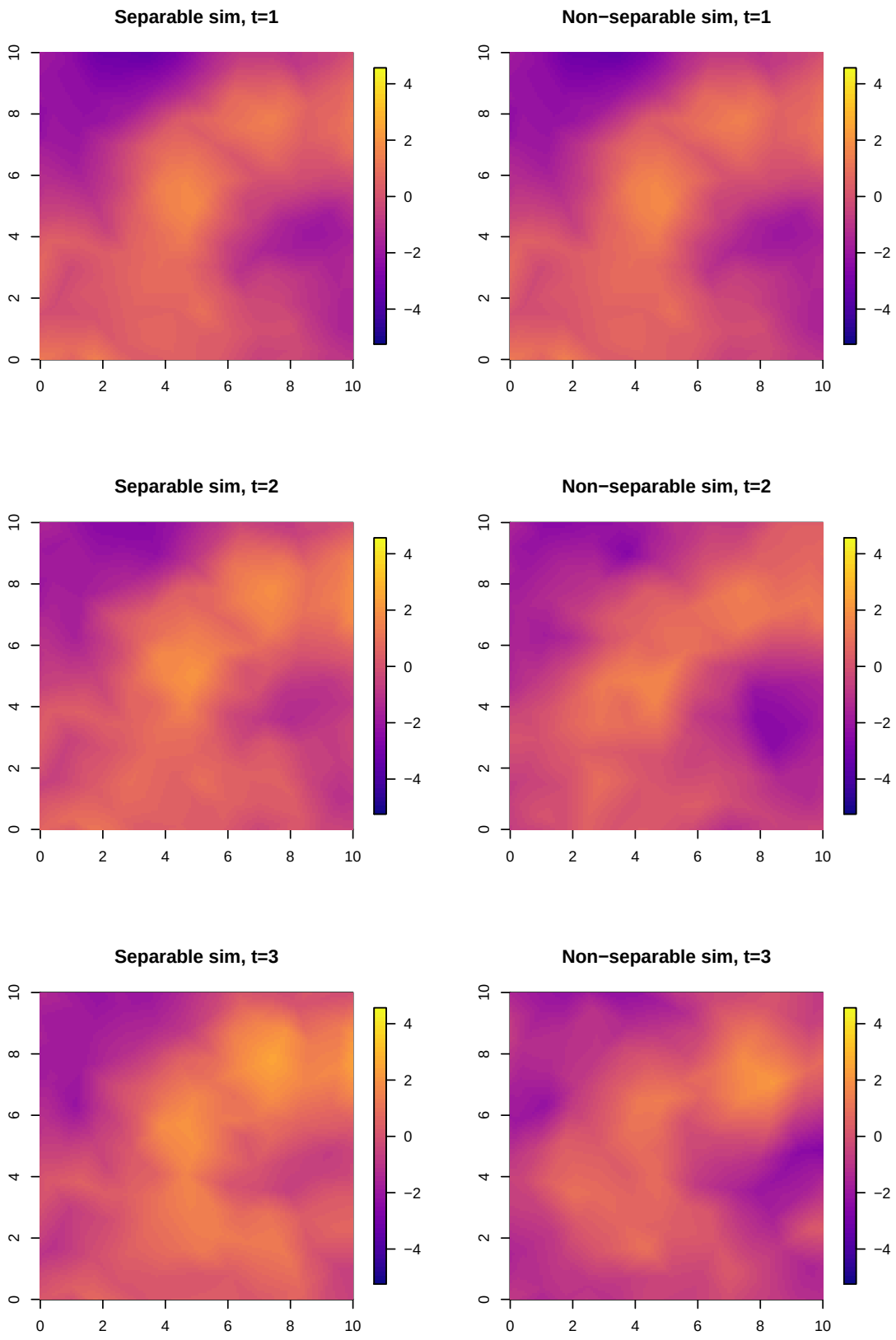

Figure 5: Posterior simulations from the separable and non-separable models.

In this code we used the option `reordering = "identity"` in the `inla.qsample` function. The purpose of this is to use the same random noise, and the same reordering, to get a close comparison between the simulations. In general, we recommend to use `inla.qsample` with a seed to get deterministic and reproducible behavior, but to use the default reordering scheme to speed up computations.

4. High performance and parallel computing with INLA

The widespread acceptance of the INLA approach and the R-**INLA** software manifested as the **INLA** package, were not foreseen when the INLA methodology was originally developed: Hence, the **INLA** package has continuously evolved from research code started more than 15 years ago, adopting designs made for single-core execution in mind. Today, there is a growing demand for analyzing much larger models: typically, either a large amount of observations and/or a large number of latent variables (read space-time models, for simplicity). And we have already started to provide better support for the increasingly larger statistical models of today running on computational platforms of tomorrow (typically multicore or manycore and possibly hardware accelerated).

At the core of the **INLA** algorithm, is numerical linear algebra for large sparse matrices. The tasks that are required, are for a symmetric positive definite matrix $\boldsymbol{Q}$ of dimension n, the ability to repeatedly compute

- the Cholesky factorization $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^\top$, where $\boldsymbol{L}$ is a lower triangular matrix,
- solve linear systems like $\boldsymbol{L}\boldsymbol{x} = \boldsymbol{b}$, $\boldsymbol{L}^\top\boldsymbol{x} = \boldsymbol{b}$, $\boldsymbol{L}\boldsymbol{L}^\top\boldsymbol{x} = \boldsymbol{b}$, and
- compute selected elements of the inverse of $\boldsymbol{Q}$, $(\boldsymbol{Q}^{-1})_{ij}$, for all ij where Q_{ij} is non-zero.

Additionally, we need also $\log|\boldsymbol{Q}|$, but since the Cholesky factor is available, it is simply $\sum_i 2\log L_{ii}$. During the entire **INLA** algorithm, the non-zero pattern of $\boldsymbol{Q}$ is the same, which simplifies some of the initial procedures, like finding a good reordering scheme.

For smaller n, like $n \sim 10^4$ to 10^5 for a spatial model, the serial algorithms for these tasks will run fine, as we have parallelized (using OpenMP) on a higher level like factorizing several matrices at once. For larger n, like $n \sim 10^5$ to 10^6, this approach is no longer practical. Also, the type of model considered plays a role here; space-time models are $\mathcal{O}(\sqrt{n})$ more costly, and require more memory, than a spatial one, hence dimension where the serial sparse matrix algorithms is no longer practical, will be less.

The need for parallel numerical methods for large sparse matrices on shared-memory and distributed-memory multiprocessors, has been evident for quite some time. While there is a vast literature on the development of efficient algorithms for the direct solution of sparse linear systems of equations, only a few software package are available, such as, e.g., **MUMPS** (Amestoy, Duff, Koster, and L'Excellent 2001; Amestoy, Guermouche, L'Excellent, and Pralet 2006), **WSMP** (Gupta 2002), **SuperLU** (Li 2005), **CHOLMOD** (Davis 2006). Neither of these libraries provide parallel algorithms for all our required matrix operations listed above, as they do not have a parallel implementation of the algorithm to compute selected elements of the inverse. (**CHOLMOD** supports a serial version of this algorithm.) How to efficiently compute selected elements of the inverse of a sparse matrix, has been known for a quite some time (Takahashi, Fagan, and Chen 1973; Erisman and Tinney 1975), but a parallel

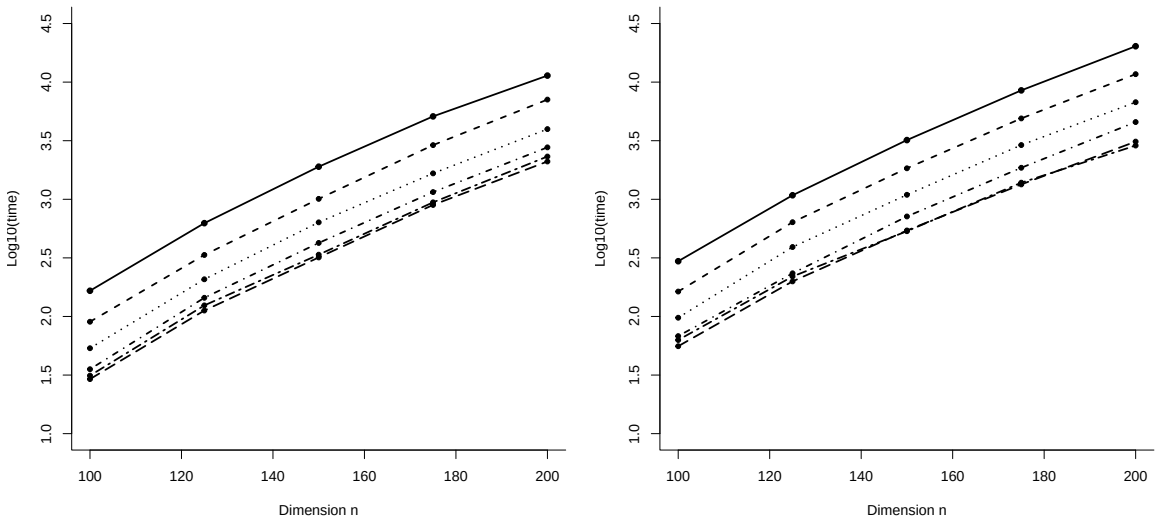

Figure 6: The running time doing Cholesky factorization (left) and computing the partial inverse for the 3D Laplace equation matrix with additional 25 dense row and columns. The dimension is $n^3 + 25$ with n vary from 100 to 200. The number of cores are 1 (top), 2, 4, 8, 12, 16 and 32 (bottom).

version of this algorithm was not available in a main sparse matrix library before the work of Verbosio, Coninck, Kourounis, and Schenk (2017) was made available in the **PARDISO** package (Schenk and Gärtner 2004; Kuzmin, Luisier, and Schenk 2013; Petra, Schenk, Lubin, and Gärtner 2014). According to Gould, Scott, and Hu (2007), **PARDISO** is one of the best performing parallel libraries for numerical computations for large sparse matrices.

A collaboration between the **PARDISO**[1] and **INLA** projects was initiated early 2018, ending up with a special version of the **PARDISO** package for **INLA** which was integrated into **INLA** and released in May 2018. With this new tool, we are now able to run successfully statistical models with n in the millions on KAUST computational servers. The paralellization strategy, that currently is supported using argument `control.compute = list(openmp.strategy = "pardiso.parallel")`, is to do one matrix at the time using a parallel algorithm to factorize, solve and compute selected entries of the inverse. The future plans for this collaboration, includes improvement of the integration with the **INLA** algorithm including nested parallelism, and also to extend the **PARDISO** interface so we can make use of more efficiently computing capabilities exploiting the parallel computing support in **PARDISO** to enable parallel distributed and accelerated execution of the main numerical tasks required in the **INLA** algorithm.

To illustrate the abilities of the **PARDISO** package to work with huge matrices, we ran a series of tests on our computational server, with 512 Gb of RAM, 2 sockets with 16 cores per socket, and with Intel Xeon Gold 6130 CPUs @2.10GHz. The test matrix is constructed to be very challenging, mimicking a large space-time model with the same non-zero structure as the 3-dimensional Laplace equation on a $n \times n \times n$ cube (which is the worst configuration). Additionally, we added 25 dense rows/columns to mimic the presence of fixed effects in the

[1]Some may be aware of a former version of **PARDISO** which has been integrated into the Intel Math Kernel Library (MKL), a library of optimized math routines for science, engineering, and financial applications.

model. For the $(n^3 + 25) \times (n^3 + 25)$ sparse matrix, have about 56 neighbors for each node. The storage required is about 0.22 Gb for $n = 100$ and 1.72 Gb for $n = 200$, to store its non-zero elements. Additionally, we need to store their (relative) location within the matrix.

Figure 6 shows the results for $n = 100, 120, 140, 160, 180$ and 200, using $nc = 1, 2, 4, 8, 12, 16$ and 32 cores, for doing Cholesky factorization (left) and the partial inverse (right). The results demonstrate a consistent behaviour for the running time both with varying n and nc. The computational cost reduces nicely from $nc = 1$ and 2 and to 4, but then the speedup fades off. We do not gain much going beyond 16 cores for this example, and the partial inverse is somewhat more expensive to compute than the Cholesky factorization. The results are very encouraging as it shows that **PARDISO** can handle sparse matrices of this size and structure without problems. The integration of **INLA** and **PARDISO** will be further improved and we are currently working on this issue.

5. Discussion

Bayesian modeling is ever present and still increasing in popularity in applied fields of science. Initially, the inference was performed using sampling-based methods like Gibbs sampling. These methods, however, are often time-consuming and computationally inefficient. From this impediment, approximate Bayesian inference approaches sprouted. (One of) The most popular non-sampling based Bayesian inference approach is the INLA methodology, facilitated through the **INLA** package. INLA is developed for the class of LGMs, that contains most well-known statistical models. Since the inception of INLA in 2009 through the seminal paper (Rue *et al.* 2009), the use of the INLA methodology has been cited more than 3000 times.

The success of INLA as a computational inferential framework for Bayesian modeling is partly attributed to the continual development and expansion of package **INLA**. As evident in this paper, relevant statistical methodology is developed and implemented incessantly in **INLA** as to provide scientists with a computational platform for state-of-the-art Bayesian models.

The specific developments presented herein address some current Bayesian modeling demands. In biomedical applications, the use of joint models for survival and longitudinal data is imperative. The efficacy of treatments as measured on multiple endpoints is a crucial step in drug design, and necessitates the use of joint modeling of the endpoints. In this paper, we presented the implementation of joint models with one survival and one longitudinal endpoint. Future developments in this field are under way and the need for a unique interface for these joint models, based on the **INLA** architecture is clear. The potential for further developments in this realm based on **INLA** is encouraging. In the flavor of joint models, the extension to spatial joint models, joint models with competing risks or recurrent events and generalized multiple endpoint modeling are some examples of models that could be implemented in **INLA** based on the approach presented herein. Multi-state models and competing risks models are also of major interest in the biomedical field, and with their implementation in **INLA** the extensions to spatial or smoothing spline random effects would be trivial.

The innovative SPDE approach for space and space-time models as used in **INLA** serves as a gateway for extensions in the field of space-time modeling. The development of a class of non-separable space-time models is motivated by current needs in the analysis of complex real space-time data, and is based on physical diffusion processes. This extension is based on the definition of a particular SPDE which is then solved using finite element methods, and con-

trasts to more common attempts at generalizing the covariance matrix or the spectrum. This approach is unique to **INLA** (within software for Bayesian modeling, as far as we know) and ensures unequivocal computational efficiency, without additional approximations, compared to other methods in the literature.

Based on the generalization to non-separable space-time models and the increasing computational demand through big data, the ability of **INLA** to perform in a high performance computing environment necessitates the development of tools available in **INLA** that can optimally facilitate the computational burden using high performance computing architecture. To this end, we present the current and future collaborative work on this front using the **PARDISO** package in conjunction with **INLA**. This project promotes the use of **INLA** to an even wider audience and ensures the applicability of **INLA** for Bayesian inference in the future.

INLA equips the user with powerful Bayesian modeling tools that are computationally efficient and relevant. The ongoing research and development of **INLA** ensures congruence to state-of-the-art statistical methodology and places the user at the vanguard of their field.

References

Alappat C, Basermann A, Bishop AR, Fehske H, Hager G, Schenk O, Thies J, Wellein G (2020). "A Recursive Algebraic Coloring Technique for Hardware-Efficient Symmetric Sparse Matrix-Vector Multiplication." *ACM Transactions on Parallel Computing*, **7**(3), 1–37. doi:10.1145/3399732.

Amestoy PR, Duff IS, Koster J, L'Excellent JY (2001). "A Fully Asynchronous Multifrontal Solver Using Distributed Dynamic Scheduling." *SIAM Journal on Matrix Analysis and Applications*, **23**(1), 15–41. doi:10.1137/s0895479899358194.

Amestoy PR, Guermouche A, L'Excellent JY, Pralet S (2006). "Hybrid Scheduling for the Parallel Solution of Linear Systems." *Parallel Computing*, **32**(2), 136–156. doi:10.1016/j.parco.2005.07.004.

Andrinopoulou ER, Eilers PHC, Takkenberg JJM, Rizopoulos D (2018). "Improved Dynamic Predictions from Joint Models of Longitudinal and Survival Data with Time-Varying Effects Using P-Splines." *Biometrics*, **74**(2), 685–693. doi:10.1111/biom.12814.

Bakka H, Krainski E, Bolin D, Rue H, Lindgren F (2020). "The Diffusion-Based Extension of the Matérn Field to Space-Time." arXiv:2006.04917 [stat.ME], URL https://arxiv.org/abs/2006.04917.

Bakka H, Rue H, Fuglstad GA, Riebler A, Bolin D, Illian J, Krainski E, Simpson D, Lindgren F (2018). "Spatial Modeling with R-**INLA**: A Review." *Wiley Interdisciplinary Reviews: Computational Statistics*, **10**(6), e1443. doi:10.1002/wics.1443.

Bollhöfer M, Eftekhari A, Scheidegger S, Schenk O (2019). "Large-Scale Sparse Inverse Covariance Matrix Estimation." *SIAM Journal on Scientific Computing*, **41**(1), A380–A401. doi:10.1137/17M1147615.

Bollhöfer M, Schenk O, Janalik R, Hamm S, Gullapalli K (2020). "State-of-the-Art Sparse Direct Solvers." In A Grama, AH Sameh (eds.), *Parallel Algorithms in Computational Science and Engineering*, pp. 3–33. Springer-Verlag, Cham. doi:10.1007/978-3-030-43736-7_1.

Braga J, Ter Braak CJF, Thuiller W, Dray S (2018). "Integrating Spatial and Phylogenetic Information in the Fourth-Corner Analysis to Test Trait-Environment Relationships." *Ecology*, **99**(12), 2667–2674. doi:10.1002/ecy.2530.

Cox DR (1972). "Regression Models and Life-Tables." *Journal of the Royal Statistical Society B*, **34**(2), 187–202. doi:10.1111/j.2517-6161.1972.tb00899.x.

Dalongeville A, Benestan L, Mouillot D, Lobreaux S, Manel S (2018). "Combining Six Genome Scan Methods to Detect Candidate Genes to Salinity in the Mediterranean Striped Red Mullet (Mullus Surmuletus)." *BMC Genomics*, **19**(1), 217. doi:10.1186/s12864-018-4579-z.

Davis TA (2006). *Direct Methods for Sparse Linear Systems (Fundamentals of Algorithms 2)*. Society for Industrial and Applied Mathematics, Philadelphia.

Erisman AM, Tinney WF (1975). "On Computing Certain Elements of the Inverse of a Sparse Matrix." *Communications of the ACM*, **18**(3), 177–179. doi:10.1145/360680.360704.

Gould NIM, Scott JA, Hu Y (2007). "A Numerical Evaluation of Sparse Direct Solvers for the Solution of Large Sparse Symmetric Linear Systems of Equations." *ACM Transactions on Mathematical Software*, **33**(2), 10. doi:10.1145/1236463.1236465.

Graetz N, Friedman J, Osgood-Zimmerman A, Burstein R, Biehl MH, Shields C, Mosser JF, Casey DC, Deshpande A, Earl L, Reiner RC, Ray SE, Fullman N, Levine AJ, Stubbs RW, Mayala BK, Longbottom J, Browne AJ, Bhatt S, Weiss DJ, Gething PW, Mokdad AH, Lim SS, Murray CJL, Gakidou E, Hay SI (2018). "Mapping Local Variation in Educational Attainment across Africa." *Nature*, **555**(7694), 48. doi:10.1038/nature25761.

Guo X, Carlin BP (2004). "Separate and Joint Modeling of Longitudinal and Event Time Data Using Standard Computer Packages." *The American Statistician*, **58**(1), 16–24. doi:10.1198/0003130042854.

Gupta A (2002). "Recent Advances in Direct Methods for Solving Unsymmetric Sparse Systems of Linear Equations." *ACM Transactions on Mathematical Software*, **28**(3), 301–324. doi:10.1145/569147.569149.

Henderson R, Diggle P, Dobson A (2000). "Joint Modelling of Longitudinal Measurements and Event Time Data." *Biostatistics*, **1**(4), 465–480. doi:10.1093/biostatistics/1.4.465.

Hu C, Sale ME (2003). "A Joint Model for Nonlinear Longitudinal Data with Informative Dropout." *Journal of Pharmacokinetics and Pharmacodynamics*, **30**(1), 83–103. doi:10.1023/a:1023249510224.

Ibrahim JG, Chu H, Chen LM (2010). "Basic Concepts and Methods for Joint Models of Longitudinal and Survival Data." *Journal of Clinical Oncology*, **28**(16), 2796. doi:10.1200/jco.2009.25.0654.

Kim S (2016). ***JointModel**: Semiparametric Joint Models for Longitudinal and Counting Processes*. R package version 1.0, URL https://CRAN.R-project.org/package=JointModel.

Kim S, Zeng D, Taylor JMG (2017). "Joint Partially Linear Model for Longitudinal Data with Informative Drop-Outs." *Biometrics*, **73**(1), 72–82. doi:10.1111/biom.12566.

Krainski ET, Gómez-Rubio V, Bakka H, Lenzi A, Castro-Camilio D, Simpson D, Lindgren F, Rue H (2019). *Advanced Spatial Modeling with Stochastic Partial Differential Equations Using R and **INLA***. Chapman & Hall/CRC, New York. doi:10.1201/9780429031892. Github version https://www.R-INLA.org/spde-book.

Kuzmin A, Luisier M, Schenk O (2013). "Fast Methods for Computing Selected Elements of the Green's Function in Massively Parallel Nanoelectronic Device Simulations." In F Wolf, B Mohr, D An Mey (eds.), *Euro-Par 2013 Parallel Processing*, pp. 533–544. Springer-Verlag, Berlin, Heidelberg.

Li XS (2005). "An Overview of **SuperLU**: Algorithms, Implementation, and User Interface." *ACM Transactions on Mathematical Software*, **31**(3), 302–325. doi:10.1145/1089014.1089017.

Lindgren F, Rue H (2015). "Bayesian Spatial Modelling with R-**INLA**." *Journal of Statistical Software*, **63**(19), 1–25. doi:10.18637/jss.v063.i19.

Lindgren F, Rue H, Lindström J (2011). "An Explicit Link between Gaussian Fields and Gaussian Markov Random Fields: The Stochastic Partial Differential Equation Approach." *Journal of the Royal Statistical Society B*, **73**(4), 423–498. doi:10.1111/j.1467-9868.2011.00777.x.

Martino S, Akerkar R, Rue H (2011). "Approximate Bayesian Inference for Survival Models." *Scandinavian Journal of Statistics*, **38**(3), 514–528. doi:10.1111/j.1467-9469.2010.00715.x.

Osgood-Zimmerman A, Millear AI, Stubbs RW, Shields C, Pickering BV, Earl L, Graetz N, Kinyoki DK, Ray SE, Bhatt S, Browne AJ, Burstein R, Cameron E, Casey DC, Deshpande A, Fullman N, Gething PW, Gibson HS, Henry NJ, Herrero M, Krause LK, Letourneau ID, Levine AJ, Liu PY, Longbottom J, Mayala BK, Mosser JF, Noor AM, Pigott DM, Piwoz EG, Rao P, Rawat R, Reiner RC, Smith DL, Weiss DJ, Wiens KE, Mokdad AH, Lim SS, Murray CJL, Kassebaum NJ, Hay SI (2018). "Mapping Child Growth Failure in Africa between 2000 and 2015." *Nature*, **555**(7694), 41. doi:10.1038/nature25760.

Petra C, Schenk O, Lubin M, Gärtner K (2014). "An Augmented Incomplete Factorization Approach for Computing the Schur Complement in Stochastic Optimization." *SIAM Journal on Scientific Computing*, **36**(2), C139–C162. doi:10.1137/130908737.

Podschwit H, Larkin N, Steel E, Cullen A, Alvarado E (2018). "Multi-Model Forecasts of Very-Large Fire Occurences during the End of the 21st Century." *Climate*, **6**(4), 100. doi:10.3390/cli6040100.

Quintero I, Jetz W (2018). "Global Elevational Diversity and Diversification of Birds." *Nature*, **555**(7695), 246. doi:10.1038/nature25794.

Ratcliffe SJ, Guo W, Ten Have TR (2004). "Joint Modeling of Longitudinal and Survival Data via a Common Frailty." *Biometrics*, **60**(4), 892–899. doi:10.1111/j.0006-341x.2004.00244.x.

R Core Team (2021). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Rizopoulos D (2016). "The R Package **JMbayes** for Fitting Joint Models for Longitudinal and Time-to-Event Data Using MCMC." *Journal of Statistical Software*, **72**(7), 1–45. doi:10.18637/jss.v072.i07.

Rodríguez de Rivera Ó, López-Quílez A, Blangiardo M (2018). "Assessing the Spatial and Spatio-Temporal Distribution of Forest Species via Bayesian Hierarchical Modeling." *Forests*, **9**(9), 573. doi:10.3390/f9090573.

Rue H, Held L (2005). *Gaussian Markov Random Fields: Theory and Applications*. Chapman & Hall/CRC.

Rue H, Martino S, Chopin N (2009). "Approximate Bayesian Inference for Latent Gaussian Models by Using Integrated Nested Laplace Approximations." *Journal of the Royal Statistical Society B*, **71**(2), 319–392. doi:10.1111/j.1467-9868.2008.00700.x.

Rue H, Martino S, Chopin N (2011). ***INLA**: Approximate Bayesian Inference Using Integrated Nested Laplace Approximations*. URL https://www.R-INLA.org/.

Rue H, Riebler A, Sørbye SH, Illian JB, Simpson DP, Lindgren FK (2017). "Bayesian Computing with **INLA**: A Review." *Annual Reviews of Statistics and Its Applications*, **4**(March), 395–421. doi:10.1146/annurev-statistics-060116-054045.

Schenk O, Gärtner K (2004). "Solving Unsymmetric Sparse Systems of Linear Equations with **PARDISO**." *Future Generation Computer Systems*, **20**(3), 475–487. doi:10.1016/j.future.2003.07.011.

Shaddick G, Thomas ML, Amini H, Broday D, Cohen A, Frostad J, Green A, Gumy S, Liu Y, Martin RV, Pruss-Ustun A, Simpson D, Van Donkelaar A, Brauer M (2018). "Data Integration for the Assessment of Population Exposure to Ambient Air Pollution for Global Burden of Disease Assessment." *Environmental Science & Technology*, **52**(16), 9069–9078. doi:10.1021/acs.est.8b02864.

Simpson D, Rue H, Riebler A, Martins TG, Sørbye SH (2017). "Penalising Model Component Complexity: A Principled, Practical Approach to Constructing Priors." *Statistical Science*, **32**(1), 1–28. doi:10.1214/16-sts576.

Stuart-Smith RD, Brown CJ, Ceccarelli DM, Edgar GJ (2018). "Ecosystem Restructuring Along the Great Barrier Reef Following Mass Coral Bleaching." *Nature*, **560**(7716), 92. doi:10.1038/s41586-018-0359-9.

Takahashi K, Fagan J, Chen MS (1973). "Formation of a Sparse Bus Impedance Matrix and Its Application to Short Circuit Study." In *8th PICA Conference Proceedings*, pp. 63–69. IEEE Power Engineering Society. Papers presented at the 1973 Power Industry Computer Application Conference in Minneapolis, Minnesota.

Van Niekerk J, Bakka H, Rue H (2019). "Joint Models as Latent Gaussian Models – Not Reinventing the Wheel." arXiv:1901.09365 [stat.ME], URL https://arxiv.org/abs/1901.09365.

Van Niekerk J, Bakka H, Rue H (2021a). "Competing Risks Joint Models Using R-**INLA**." *Statistical Modelling*. doi:10.1177/1471082x19913654. Forthcoming.

Van Niekerk J, Bakka H, Rue H (2021b). "Principled Distance-Based Prior for the Shape of the Weibull Model." *Statistics and Probability Letters*, **174**, 109098. doi:10.1016/j.spl.2021.109098.

Verbosio F, Coninck AD, Kourounis D, Schenk O (2017). "Enhancing the Scalability of Selected Inversion Factorization Algorithms in Genomic Prediction." *Journal of Computational Science*, **22**, 99–108. doi:10.1016/j.jocs.2017.08.013.

Wulfsohn MS, Tsiatis AA (1997). "A Joint Model for Survival and Longitudinal Data Measured with Error." *Biometrics*, **53**(1), 330–339. doi:10.2307/2533118.

Zhou H, Lawson AB, Hebert JR, Slate EH, Hill EG (2008). "Joint Spatial Survival Modeling for the Age at Diagnosis and the Vital Outcome of Prostate Cancer." *Statistics in Medicine*, **27**(18), 3612–3628. doi:10.1002/sim.3232.

Affiliation:

Janet van Niekerk, Haakon Bakka, Håvard Rue
CEMSE Division
King Abdullah University of Science and Technology (KAUST)
Thuwal, Saudi Arabia
E-mail: Janet.vanNiekerk@kaust.edu.sa, Haakon.Bakka@kaust.edu.sa
Haavard.Rue@kaust.edu.sa

Olaf Schenk
Institute of Computational Science
Universita della Svizzera Italiana
Switzerland

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
November 2021, Volume 100, Issue 2 *Submitted:* 2019-06-15
doi:10.18637/jss.v100.i02 *Accepted:* 2020-09-08