New frontiers in Bayesian modeling using the INLA package in R

Janet van Niekerk, Haakon Bakka, Håvard Rue and Olaf Schenk
KAUST and USI

July 26, 2019

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.

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 in R. This package is based on the INLA methodology developed by [Rue et al. (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 et al. (2018) investigated environmental relationships by incorporating phylogenetic information. Dalongeville et al. (2018) used R-INLA to genes specific to salinity in the field of genomics (also see ). Air pollution was assessed with the purpose of disease assessment by Shaddick et al. (2018) while Rodríguez de Rivera et al. (2018) used R-INLA to determine forest species distributions. A study into fire occurrences was conducted by Podschweit et al. (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 et al. (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 pertinence 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. Latent Gaussian models is a specific subset of hierarchical Bayesian additive models. This class comprises of 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), \( \theta \) and the linear predictors, \( \eta_i \), such that the complete likelihood can be expressed as

\[
\pi(y|\eta, \theta) = \prod_{i=1}^{N} \pi(y_i|\eta_i(\mathbf{X}), \theta).
\]

The linear predictor is formulated as follows:

\[
\eta_i = \beta_0 + \beta^T \mathbf{X}_i + \mathbf{u}_i(z_i)
\]

where \( \beta \) represent the linear fixed effects of the covariates \( \mathbf{X} \), \( \epsilon \) is the unstructured random effects and the unknown non-linear functions \( \mathbf{u} \) of the covariates \( z \) 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 include 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, \( Y \) is conditionally independent given the partially observed latent field, \( \mathbf{X} \) and some hyper parameters \( \theta_1 \). The latent field \( \mathbf{X} \) is formed from the structured predictor as \( (\beta, \mathbf{u}, \eta) \) which forms a Gaussian Markov random field with sparse precision matrix \( Q(\theta_2) \), i.e. \( \mathbf{X} \sim N(0, Q^{-1}(\theta_2)) \). A prior, \( \pi(\theta) \) can then be formulated for the set of hyper parameters \( \theta = (\theta_1, \theta_2) \). The joint posterior distribution is then given by:

\[
\pi(\mathbf{X}, \theta) \propto \pi(\theta)\pi(\mathbf{X}|\theta) \prod_i \pi(Y_i|\mathbf{X}, \theta)
\]

The goal is to approximate the joint posterior density \( \pi(\mathbf{X}, \theta) \) and subsequently compute the marginal posterior densities, \( \pi(\mathbf{X}|Y), i = 1...n \) and \( \pi(\theta|Y) \). 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. 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. Finally, we discuss how the INLA package is adapted to a high performance computing environment using the PARDISO library.

2 Complex survival models using the INLA package

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 nonparametric 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)).

2.1 Joint longitudinal-survival models

A 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 et al. (2008); Ratcliffe et al. (2004); Andrinopoulou et al. (2018)). Each of these new joint models is still a latent Gaussian model and thus no special implementation package is needed for each one (for more details see Van Niekerk et al. (2019)). Most longitudinal likelihoods and hazard function assumptions can be facilitated in this framework, leaving no need to develop a new implementation for each set of assumptions.

Within the realm of joint longitudinal-survival models, users have a choice of various computational approaches. The joineR library in R is widely used to fit joint models from a frequentist point of view whereas the JMBayes library facilitates Bayesian estimation of joint models. The joineR library can even accommodate competing events in the survival submodel. In terms of partially linear joint models the JointModel library was developed to fit non-linear covariate effects in the longitudinal submodel using B-splines with a sieve approximation. The bamls library can also be used to fit a partially linear joint model using a Markov Chain Monte Carlo (MCMC) approach. We, however, aim to show that most joint models (also with non-linear covariate effects) can be fitted using the INLA library, also including discrete and non-Gaussian continuous joint models. This provides the user with one computational tool for the Bayesian inference of most joint models, since our approach provides support for non-linear covariate
effects through continuously-indexed splines as well as discrete and continuous spatial random effects.

2.1.1 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 sub models, and then focus on the joint model in its entirety. Suppose the hazard rate for individual \( i, i = 1, ..., N^S \) 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, 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

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

then the likelihood for the survival sub model is

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

(3)

where \( c_i = I(\text{non-censored observation}) \) 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 and can be accommodated in our approach. The observations are thus a mixture of event times and censored times, dependent on the status of each individual.

Now, for the longitudinal data suppose that each individual has \( N_i, i = 1, ..., N^S \) observations \( y_{ij}, i = 1, ..., N^S, j = 1, ..., N_i \) for a total longitudinal data set size of \( N^L = \sum_{i=1}^{N^S} N_i \). We specify the linear predictor \( \eta_i^L(t) \), based on covariates at time \( t \), and a conditional density function \( g(y_i|\eta_i^L(t)) \) for individual \( i \) resulting in the likelihood for the longitudinal sub model as

\[
\pi_L(y|\eta^L) = \prod_{i=1}^{N_L} g(y_i|\eta_i^L(t)).
\]

(4)
Now consider the linear predictors of the joint model,

\[
\begin{align*}
\eta_{L,J}^L(t) &= \eta_L^L(t) \\
\eta_{S,J}^S(s) &= \eta_S^S(s) + f(\eta_L^L(s)),
\end{align*}
\]

where \( \eta^S \) and \( \eta^L \) are of the form (1) and \( f: \mathbb{R} \to \mathbb{R} \) is a smooth function of \( \eta_L^L(t) \). The function \( f \) 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 et al. (2010)), while traditionally only the subject-specific intercept and slope of the time effect have been used i.e. \( f(\eta_L^L(s)) = \nu_1 w_1 + \nu_2 w_2 s \). In the latter we assume the structure specified by Henderson et al. (2000) as follows,

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

Note that either \( \nu_1 \) or \( \nu_2 \) 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 where \( f(\eta_L^L(s)) = \nu_1 w_1 + \nu_2 w_2 s \). Secondly we present an example with a non-linear trajectory and informative dropout event process with \( f(\eta_L^L(s)) = \nu \eta_L^L(s) \) from a prostate cancer study using post treatment PSA levels as a longitudinal bio marker.

**Example 1. HIV antiretroviral treatments efficacy**

In this example the efficacy and safety of two antiretroviral treatments, Didanosine and Zalcitabine, is investigated and presented in Guo and Carlin (2004). This randomized trial includes only patients who had failed or were intolerant to Zidovudine (AZT) therapy. In the joint model, we use the same association structure as in Guo and Carlin (2004), i.e.

\[
\begin{align*}
\eta_{L,J}^L(t) &= \eta_L^L(t) + w_1 + w_2 t \\
\eta_{S,J}^S(s) &= \eta_S^S(s) + \nu_1 w_1 + \nu_2 w_2 s.
\end{align*}
\]

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 sub models. The specific sub models are then

\[
\begin{align*}
\eta_{L,J}^L(t) &= \beta_0^L + \beta_1^L \text{Gender} + \beta_2^L \text{Drug} + \beta_3^L \text{Previous OI} + \beta_4^L \text{AZT Resistance} + w_1 + w_2 t \\
\eta_{S,J}^S(s) &= \beta_0^S + \beta_1^S \text{Gender} + \beta_2^S \text{Drug} + \beta_3^S \text{Previous OI} + \beta_4^S \text{AZT Resistance} + \nu_1 w_1 + \nu_2 w_2 s.
\end{align*}
\]

The data is loaded and visualized by
R> library("INLA")
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$CD4[1:100], + xlab="Time(years)", ylab="CD4 count", + legend=F, col=c(1:467))
R> hist(aids$CD4, main="",xlab="CD4 count")

Figure 1: Individual profiles and histogram of CD4 counts

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 sub model. 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(max(data1$Time),max(data1$obstime))
R> mtime

[1] 21.4
All the times hereafter should thus be rescaled to (0; 21.4) for interpretation. Some preprocessing of the data is required to perform the joint analysis. The details are omitted here but the concept is illustrated in [8]. Define,

\[
y = \begin{bmatrix}
y_1 & \text{NA} \\
y_2 & \text{NA} \\
\vdots & \vdots \\
y_{N_L} & \text{NA} \\
\text{NA} & s_1 \\
\text{NA} & s_2 \\
\vdots & \vdots \\
\text{NA} & s_N \\
\end{bmatrix}
\]

\[
\beta = [\beta^L_1, \beta^S_1, \ldots]
\]

\[
X = \begin{bmatrix}
x_{1,1}^L & 0 & \ldots \\
x_{1,2}^L & 0 & \ldots \\
\vdots & \vdots & \ddots \\
x_{1,N_L}^L & 0 & \ldots \\
0 & x_{1,1}^S & \ldots \\
0 & x_{1,2}^S & \ldots \\
0 & \ldots & \ldots \\
0 & x_{1,N} & \ldots \\
\end{bmatrix}
\]

\[
u(t) = \begin{bmatrix}
w_{1,1} & w_{2,1} & \text{NA} & \text{NA} \\
w_{1,1} & w_{2,1} & \text{NA} & \text{NA} \\
\vdots & \vdots & \vdots & \vdots \\
w_{1,N} & w_{2,N} & \text{NA} & \text{NA} \\
\text{NA} & \text{NA} & \nu_1 w_{1,1} & \nu_2 w_{2,1} s_1 \\
\text{NA} & \text{NA} & \nu_1 w_{1,2} & \nu_2 w_{2,2} s_2 \\
\vdots & \vdots & \vdots & \vdots \\
\text{NA} & \text{NA} & \nu_1 w_{1,N} & \nu_2 w_{2,N} s_N \\
\end{bmatrix}
\]

Then the joint model in (7) is an LGM similar to (1). In this paper, we use the pre-processed data available in the INLA package using the following code.

```R
joint.dataCD4 = readRDS(system.file("exampledata/cd4/jointdataCD4.rds", + package = "INLA"))
```

The joint model is fitted 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 distribution for the hazard rate.

```R
JointmodelCD4 = inla(Y ~ -1 + mu + l.gender + l.drug + l.prevOI + + 1.AZT + s.gender + s.drug + s.prevOI + s.AZT +
```

+ f(U11, model="iid2d", n=2*length(joint.dataCD4$mu)) +
+ 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))

R> summary(JointmodelCD4)

Fixed effects:

|       | mean | sd  | 0.025 | 0.5quant | 0.975quant | mode | kld |
|-------|------|-----|-------|----------|------------|------|-----|
| mu1   | 2.336| 0.095| 2.150 | 2.336    | 2.524      | 2.336|     |
| mu2   | -1.171| 0.296| -1.777| -1.162   | -0.616     | -1.145|     |
| l.gender2 | -0.016 | 0.091 | -0.195 | -0.017 | 0.163 | -0.017 |     |
| l.drug2    | 0.059 | 0.053 | -0.046 | 0.059 | 0.163 | 0.059 |     |
| l.prevOI2  | -0.692 | 0.067 | -0.824 | -0.692 | -0.560 | -0.691 |     |
| l.AZT2     | -0.020 | 0.070 | -0.157 | -0.020 | 0.117 | -0.020 |     |
| s.gender2  | -0.340 | 0.247 | -0.801 | -0.348 | 0.169 | -0.365 |     |
| s.drug2    | 0.211 | 0.148 | -0.078 | 0.211 | 0.502 | 0.211 |     |
| s.prevOI2  | 1.286 | 0.228 | 0.849 | 1.282 | 1.745 | 1.274 |     |
| s.AZT2     | 0.154 | 0.164 | -0.166 | 0.153 | 0.479 | 0.151 |     |

Model hyperparameters:

|                                | mean | sd  | 0.025quant | 0.5quant |
|--------------------------------|------|-----|------------|----------|
| alpha parameter for weibullsurv[2] | 1.279| 0.056| 1.169      |          |
| Precision for U11 (component 1)  | 4.233| 0.416| 3.477      |          |
| Precision for U11 (component 2)  | 3.811| 2.517| 0.777      |          |
| Rho1:2 for U11                   | 0.095| 0.305| -0.480     |          |
| Beta for U12                      | -1.048| 0.218| -1.473     |          |
| Beta for U22                      | 0.977| 0.281| 0.418      |          |
Similarly to Guo and Carlin (2004), the status of previous opportunistic infection (prevOI) is a significant covariate in both the longitudinal and survival models. The association between the longitudinal and survival models is significant. Entry into the study with a previous AIDS diagnosis results in lower CD4 counts and an increased hazard of death. There is a negative significant association between the initial value of the CD4 trajectory and the linear predictor of the survival model, which indicates a decreased hazard of death for individuals with higher CD4 counts at study entry. The positive association between the hazard rate and the slope of CD4 is an anomaly which is explained by the use of a Weibull survival model with an estimated shape parameter of 1.398, which indicates an increase in hazard over time. The random time trend association we aim to capture in $\nu_2$ is thus construed with the shape parameter of the Weibull model.

To use the model for patient-specific predictions we extract the necessary components from the latent field of the longitudinal and survival sub models. We use the data in dataH to calculate the survival functions and dataL1 to illustrate the observed and estimated longitudinal trajectories.

```r
data1$Time = data1$Time/mtime
data1$obstime = data1$obstime/mtime
data = data1[dat1$obstime==0,]
data1 = data1[, c(1,4:12)]
ns = nrow(data)
nl = nrow(data1)
dataH = data.frame(data,
+ int.re = JointmodelCD4$summary.random$U12$mean[(nl+1):(nl+ns)],
+ slope.re = JointmodelCD4$summary.random$U22$mean[(nl+1):(nl+ns)])
dataL1 = data.frame(data1,
+ fitted.l = JointmodelCD4$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 produce the patient-specific CD4 trajectories and survival curves for two patients, one with AIDS infection at entry (patient 4) and one without (patient 35) in Figure 2.
R> patients = c(4,35)
R> par(mfrow = c(2,2))
R> par(mar = c(4,4,4,4))
R> for (patientnr in patients){
+   dataHi = dataH[dataH$patient==patientnr,]
+   lambda = exp(-1.171 + 1.274 * (as.numeric(as.factor(dataHi$prevOI))
+                     - 1) - 1.053 * dataHi$int_re)
+   alpha=1.282
+   plot(dataL$obstime[dataL$patient==patientnr] * 21.4,
+        dataL$CD4[dataL$patient==patientnr],ylab = "CD4 count",
+        xlab = "Time (months)",type = "l",xlim = c(0,21.4),
+        ylim = c(0,15),main = paste("CD4 trajectory - patient",patientnr))
+   lines(dataL$obstime[dataL$patient==patientnr] * 21.4,
+         (dataL$fitted_l[dataL$patient==patientnr]
+          + dataL$random_l[dataL$patient==patientnr]
+          + dataL$randoms_l[dataL$patient==patientnr]),
+         col="blue",lty=2)
+   plot(seq(0.1,1,0.1) * 21.4 * 5, exp(-(seq(0.1,1,0.1) * 5) ^ alpha)
+        * (lambda + 0.993 * dataHi$slope_re * 5 * seq(0.1,1,0.1))),
+        type = "l",ylab = "Survival probability",xlab = "Time (months)",
+        main = paste("Survival curve - patient",patientnr))
+   abline(h = 0.5, col = "red")
+ }

10
Example 2. PSA levels and informative dropout

We follow [Hu and Sale (2003) and Kim et al. (2017)] to estimate the longitudinal trajectory by correcting for the bias introduced by informative dropout. Since the main 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. $f(\eta^L_i(s)) = \nu \eta^L_i(s)$. To model the non-linear trajectory we use a random walk order two component over time $\alpha(t)$. The model is thus:

$$
\eta^L_i(t) = \beta^L_0 + \alpha(t) + \beta^L_t PSA_{base}
$$

$$
\eta^S_i(s) = \beta^S_0 + \nu \eta^L_i(s),
$$

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".

```r
library("JointModel")
inla.setOption(short.summary = TRUE)
data1 = prostate
data2 = dropout
ng = nrow(data1)
ns = nrow(data2)
data1 = data1[order(data1$VisitTime),]
joint.dataPSA = readRDS(system.file("exampledata/psa/jointdataPSA.rds", + package="INLA"))
Jointmodelres1 = inla(Y ~ -1 + mu + 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","weibullsurb"), data = + joint.dataPSA, verbose=TRUE, control.compute=list + (dic = TRUE,config = TRUE), control.family = + list( (), + list(hyper = list(prec = list(initial = 10, + fixed = TRUE)) ), list( ()) )
summary(Jointmodelres1)
```

**Fixed effects:**

|        | mean   | sd    | 0.025quant | 0.5quant | 0.975quant |
|--------|--------|-------|------------|----------|------------|
| mode kld |        |       |            |          |            |
| mu1    | 0.083  | 0.038 | 0.008      | 0.083    | 0.157      |
| 0      |        |       |            |          |            |
| mu2    | -0.987 | 0.185 | -1.366     | -0.982   | -0.640     |
| 0      |        |       |            |          |            |
| b13.PSABase | 0.421  | 0.026 | 0.370      | 0.421    | 0.472      |
| 0      |        |       |            |          |            |

**Model hyperparameters:**

|      | mean   | sd   | 0.025quant |
|------|--------|------|------------|
|      |        |      |            |

12
The association parameter $\nu$ is significant and it is thus clear that the joint model approach is necessary for this data set. The shape parameter for the Weibull model is estimated at 0.7863 which is different from 1 and thus the exponential model would not suffice. The estimated non-linear longitudinal average trajectory is illustrated in Figure 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 et al. (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 et al. (2019) 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

$$L = (\gamma^2 + \Delta),$$

Figure 3: Average PSA trajectory (red line) and estimated PSA levels (blue dots)
where \( \gamma_s^2 \) 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 net transport from high value regions to low value regions (e.g. temperature equalises over time).

There is a rich literature on space-time models with non-separable covariance functions, see e.g. Gneiting (2002), Stein (2013), Rodrigues and Diggle (2010), and references therein. The literature focuses on constructing a model with a reasonable covariance function, and developing computationally efficient method for inference. However, as far as we are aware, there are no flexible software implementations ready for use with these models.

In this section we discuss an implementation of the new models using the computational methods in INLA, by writing R code for the inference explicitly instead of using the \texttt{inla} function call. This is both to reduce the computational time for this case study and to be transparent with all the details. We recommend the reader to consider 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.

\begin{verbatim}
R> library("fields")
R> library("colospace")
R> set.seed(2019)

We use simple temporal and spatial meshes as follows. The spatial mesh can be plotted by \texttt{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 = T)
R> mesh.space = inla.mesh.2d(loc = fake.locations, max.edge=c(1.5, 2))

We use \( \alpha_t = 1, \alpha_s = 2, \alpha_\epsilon = 1 \) in the model in Bakka et al. (2019), and get the SPDE

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

where the \( \gamma \)'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 use the empirical range for space and time from Lindgren et al. (2011). The hyper-parameters for the separable model are as follows.

R> range.time = 20
R> range.space = 6
R> sigma.u = 1
We select hyper-parameters (γ’s) for the non-separable model to have a similar interpretation to the interpretation of the parameters of the separable model [Bakka et al. 2019].

R> gt = 2.23
R> gs2 = 8/range.space^2
R> ge2 = 0.0805

We use the finite element method (FEM) from [Lindgren et al. 2011], adopted by [Bakka et al. 2019] to the following $M$-notation. We note that the temporal model is first order Markov, and that higher order Markov structure is used for models with a higher smoothness in time [Bakka et al. 2019].

R> sfe = inla.mesh.fem(mesh.space, order = 4)
R> tfe = inla.mesh.fem(mesh.time, order = 2)
R> M0 = tfe$c0
R> stopifnot(abs(M0[1,1] - 0.5*M0[2,2])<1e-3)
R> N.t = nrow(M0)
R> M1 = sparseMatrix(i=c(1,N.t), j=c(1,N.t), x=0.5)
R> M2 = tfe$g1

The sfe and tfe objects contain the finite element matrices we need to build a solution to the SPDE. 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.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 and covariance structure as follows. Importantly, we note that the marginal variance is close to 1 for all models.

R> print(diag(inla.qinv(Q.M)))
R> print(solve(Q.M)[1,])
R> print(summary(diag(inla.qinv(Q.separ))))
R> print(summary(diag(inla.qinv(Q.nonsep))))

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.
We follow the book Rue and Held (2005) and compute posterior precision matrices and means, by conditioning on $y$. Here, $Q_{\text{eps}}$ is the precision matrix for the observation noise, and $A_{\text{observe}}$ is the matrix projecting from the latent field to the observation locations.

```r
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> ## Posterior/conditional precision matrix
R> post.Q = function(sig.eps=0.01, sig.v=1, Q.model) {
+   Q = sig.eps^(-2) * t(A.observe)%*%Qeps%*%A.observe + Q.model
+   return(Q)
+ }
R> ## Posterior mean (point estimate)
R> post.mu = function(sig.eps=0.01, sig.v=1, 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, sig.v, 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
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 = diverging_hcl(63), ...)
+ }
```

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 to 6, however, represent forecasts based on the data observed in year 1. The plots shown here are for the first three years, 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 of the current observations, while the non-separable model results in smoother forecasts. 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
R> par(mfrow = c(3, 2))
R> zlim2 = c(-1, 1)*max(abs(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)
+ }
```
Figure 4: Posterior mean estimates from the separable and non-separable models.
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.

```r
post.sim.separ = inla.qsample(1, Q=post.Q(Q.model = Q.separ),
                             reordering = "identity", seed = 1,
                             + num.threads = 1)
post.sim.separ = drop(post.sim.separ + mu.post.separ)
post.sim.nonsep = inla.qsample(1, Q=post.Q(Q.model = Q.nonsep),
                               reordering = "identity", seed = 1,
                             + num.threads = 1)
post.sim.nonsep = drop(post.sim.nonsep + mu.post.nonsep)
zlim1 = c(-1, 1) * max(abs(c(post.sim.separ, post.sim.nonsep)))
par(mfrow = c(3, 2))
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)
}
```

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 behaviour, but to use the default reordering scheme to speed up computations.
Figure 5: Posterior simulations from the separable and non-separable models.
4 High performance and parallel computing with the INLA package

The widespread acceptance of the INLA-approach and the RINLA software manifested as the INLA package, was not foreseen when INLA 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 analysing 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 is required, are for a symmetric positive definite matrix \( Q \) of dimension \( n \), the ability to repeatedly compute

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

Additionally, we need also \( \log|Q| \), but since the Cholesky factor is available, it is simply \( \sum_i 2 \log L_{ii} \). During the whole INLA algorithm, the non-zero pattern of \( Q \) 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 \( n \sim 10^5 \) to \( 10^6 \), this approach is no longer practical. Also, the type of model considered plays a role here; space-time models is \( \mathcal{O}(\sqrt{n}) \) 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, have 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 et al., 2001, 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 support a serial version of this algorithm.) How to efficiently compute selected elements of the inverse of a sparse matrix, have been known for a quite some time (Takahashi et al., 1973; Erisman and Tinney, 1975), but a parallel version
of this algorithm was not available in a main sparse matrix library before the work of Verbosio et al. (2017) was made available in the PARDISO library (Schenk and Gärtner, 2004; Kuzmin et al., 2013; Petra et al., 2014). According to Gould et al. (2007), PARDISO one of the best performing parallel libraries for numerical computations for large sparse matrices.

A collaboration between PARDISO and INLA project was initiated early 2018, ending up with a special version of the PARDISO library 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 parallelisation 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 library to work with huge matrices, we ran a series of tests our computational server, with 512Gb of RAM, 2 sockets with 16 cores per socket, and with Intel(R) Xeon(R) 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 model. For the \( (n^3 + 25) \times (n^3 + 25) \) sparse matrix, have about 56 neighbors for each node. The storage required is about 0.22Gb for \( n = 100 \) and 1.72Gb 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.

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.

1www.pardiso-project.org
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).

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. Since the inception of INLA in 2009 through the seminal paper [10], the use of the INLA methodology has
been cited over 3000 times. INLA is developed for the class of latent Gaussian models, that contains most well-known statistical models.

The success of INLA as a computational inferential framework for Bayesian modeling is partly attributed to the continual development and expansion of this package. 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. Multistate 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 contrasts 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 library 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

Amestoy, P. R., Duff, I. S., Koster, J., and L’Excellent, J.-Y. (2001). A fully asynchronous multifrontal solver using distributed dynamic scheduling. *SIAM Journal on Matrix Analysis and Applications*, 23(1):15–41.

Amestoy, P. R., Guermouche, A., L’Excellent, J.-Y., and Pralet, S. (2006). Hybrid scheduling for the parallel solution of linear systems. *Parallel Computing*, 32(2):136–156.

Andrinopoulou, E.-R., Eilers, P. H., Takkenberg, J. J., and 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.

Bakka, H., Krainski, E., Bolin, D., Rue, H., and Lindgren, F. (2019). A class of interpretable and computationally efficient non-separable space-time models based on stochastic partial differential equations. *arXiv preprint arXiv:xxx.xxx*. Work in progress (reference will be updated).

Bakka, H., Rue, H., Fuglstad, G.-A., Riebler, A., Bolin, D., Illian, J., Krainski, E., Simpson, D., and Lindgren, F. (2018). Spatial modeling with R-INLA: A review. *Wiley Interdisciplinary Reviews: Computational Statistics*, 10(6):e1443.

Braga, J., ter Braak, C. J., Thuiller, W., and Dray, S. (2018). Integrating spatial and phylogenetic information in the fourth-corner analysis to test trait–environment relationships. *Ecology*, 99(12):2667–2674.

Cox, D. R. (1972). Regression models and life-tables. *Journal of the Royal Statistical Society: Series B (Methodological)*, 34(2):187–202.

Dalongeville, A., Benestan, L., Mouillot, D., Lobreaux, S., and 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.

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

Erisman, A. M. and Tinney, W. F. (1975). On computing certain elements of the inverse of a sparse matrix. *Communications of the ACM*, 18(3):177–179.

Gneiting, T. (2002). Nonseparable, stationary covariance functions for space–time data. *Journal of the American Statistical Association*, 97(458):590–600.

Gould, N. I. M., Scott, J. A., and Hu, Y. (2007). A numerical evaluation of sparse direct solvers for the solution of large sparse symmetric linear systems of equations. *ACM Trans. Math. Softw.*, 33(2).
Graetz, N., Friedman, J., Osgood-Zimmerman, A., Burstein, R., Biehl, M. H., Shields, C., Mosser, J. F., Casey, D. C., Deshpande, A., Earl, L., et al. (2018). Mapping local variation in educational attainment across Africa. *Nature*, 555(7694):48.

Guo, X. and Carlin, B. P. (2004). Separate and joint modeling of longitudinal and event time data using standard computer packages. *The American Statistician*, 58(1):16–24.

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.

Henderson, R., Diggle, P., and Dobson, A. (2000). Joint modelling of longitudinal measurements and event time data. *Biostatistics*, 1(4):465–480.

Hu, C. and Sale, M. E. (2003). A joint model for nonlinear longitudinal data with informative dropout. *Journal of Pharmacokinetics and Pharmacodynamics*, 30(1):83–103.

Ibrahim, J. G., Chu, H., and Chen, L. M. (2010). Basic concepts and methods for joint models of longitudinal and survival data. *Journal of Clinical Oncology*, 28(16):2796.

Kim, S., Zeng, D., and Taylor, J. M. (2017). Joint partially linear model for longitudinal data with informative drop-outs. *Biometrics*, 73(1):72–82.

Krainski, E. T., Gómez-Rubio, V., Bakka, H., Lenzi, A., Castro-Camilo, D., Simpson, D., Lindgren, F., and Rue, H. (2019). *Advanced Spatial Modeling with Stochastic Partial Differential Equations using R and INLA*. New York: Chapman and Hall/CRC. Github version www.r-inla.org/spde-book.

Kuzmin, A., Luisier, M., and Schenk, O. (2013). Fast methods for computing selected elements of the green’s function in massively parallel nanoelectronic device simulations. In Wolf, F., Mohr, B., and an Mey, D., editors, *Euro-Par 2013 Parallel Processing*, pages 533–544, Berlin, Heidelberg. Springer Berlin Heidelberg.

Li, X. S. (2005). An overview of SuperLU: Algorithms, implementation, and user interface. *ACM Trans. Math. Softw.*, 31(3):302–325.

Lindgren, F., Rue, H., and 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.

Martino, S., Akerkar, R., and Rue, H. (2011). Approximate Bayesian inference for survival models. *Scandinavian Journal of Statistics*, 38(3):514–528.
Osgood-Zimmerman, A., Millear, A. I., Stubbs, R. W., Shields, C., Pickering, B. V., Earl, L., Graetz, N., Kinyoki, D. K., Ray, S. E., Bhatt, S., et al. (2018). Mapping child growth failure in Africa between 2000 and 2015. *Nature*, 555(7694):41.

Petra, C., Schenk, O., Lubin, M., and GÃ¤rtner, K. (2014). An augmented incomplete factorization approach for computing the Schur complement in stochastic optimization. *SIAM Journal on Scientific Computing*, 36(2):C139–C162.

Podschwit, H., Larkin, N., Steel, E., Cullen, A., and Alvarado, E. (2018). Multimodel forecasts of very-large fire occurrences during the end of the 21st century. *Climate*, 6(4):100.

Quintero, I. and Jetz, W. (2018). Global elevational diversity and diversification of birds. *Nature*, 555(7695):246.

Ratcliffe, S. J., Guo, W., and Ten Have, T. R. (2004). Joint modeling of longitudinal and survival data via a common frailty. *Biometrics*, 60(4):892–899.

Rodrigues, A. and Diggle, P. J. (2010). A class of convolution-based models for spatio-temporal processes with non-separable covariance structure. *Scandinavian Journal of Statistics*, 37(4):553–567.

Rodríguez de Rivera, Ó., López-Quílez, A., and Blangiardo, M. (2018). Assessing the spatial and spatio-temporal distribution of forest species via bayesian hierarchical modeling. *Forests*, 9(9):573.

Rue, H. and Held, L. (2005). *Gaussian Markov random fields: theory and applications*. CRC press.

Rue, H., Martino, S., and 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.

Schenk, O. and Gärtnner, K. (2004). Solving unsymmetric sparse systems of linear equations with PARDISO. *Future Generation Computer Systems*, 20(3):475 – 487. Cited>1200.

Shaddick, G., Thomas, M. L., Amini, H., Broday, D., Cohen, A., Frostad, J., Green, A., Guny, S., Liu, Y., Martin, R. V., et al. (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.

Stein, M. L. (2013). On a class of space–time intrinsic random functions. *Bernoulli*, 19(2):387–408.

Stuart-Smith, R. D., Brown, C. J., Ceccarelli, D. M., and Edgar, G. J. (2018). Ecosystem restructuring along the great barrier reef following mass coral bleaching. *Nature*, 560(7716):92.
Takahashi, K., Fagan, J., and Chen, M. S. (1973). Formation of a sparse bus impedance matrix and its application to short circuit study. In 8th PICA Conference proceedings, pages 63–69. IEEE Power Engineering Society. Papers presented at the 1973 Power Industry Computer Application Conference in Minneapolis, Minnesota.

Van Niekerk, J., Bakka, H., and Rue, H. (2019). Joint models as latent gaussian models—not reinventing the wheel. arXiv preprint arXiv:1901.09365.

Verbosio, F., Coninck, A. D., Kourounis, D., and Schenk, O. (2017). Enhancing the scalability of selected inversion factorization algorithms in genomic prediction. Journal of Computational Science, 22:99 – 108.

Wulfsohn, M. S. and Tsiatis, A. A. (1997). A joint model for survival and longitudinal data measured with error. Biometrics, pages 330–339.

Zhou, H., Lawson, A. B., Hebert, J. R., Slate, E. H., and Hill, E. G. (2008). Joint spatial survival modeling for the age at diagnosis and the vital outcome of prostate cancer. Statistics in medicine, 27(18):3612–3628.