Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. *doi: 10.18637/jss.v000.i00*

Modelling, Fitting, and Prediction with Non-Gaussian Spatial and Spatio-Temporal Data using TMB and FRK

Matthew Sainsbury-Dale University of Wollongong

Andrew Zammit-Mangion University of Wollongong

Noel Cressie University of Wollongong

Abstract

FRK is an R package for spatial/spatio-temporal modelling and prediction with large data sets. In this paper, we describe extensions to **FRK** that allow for modelling with non-Gaussian data and using as an increased number of basis functions in the spatial random process. We employ the generalised linear mixed model framework, integrating out the latent random effects using an implementation of the Laplace approximation provided by the R package **TMB**. The existing functionality of **FRK** is retained with this upgrade; in particular, it makes use of automatic basis function construction, it is capable of handling both point-referenced and areal data simultaneously, and it eases the so-called spatial change-of-support problem. We demonstrate new features in **FRK** and compare it to alternative models, using both simulated and real data sets.

Keywords: non-Gaussian spatial and spatio-temporal data, basis functions, R, change of support, areal data.

1. Introduction

FRK, introduced by Zammit-Mangion and Cressie (2021), is an R (R Core Team 2020) package for spatial/spatio-temporal modelling and prediction. It can handle large point- and areally-referenced data sets, and seamlessly deals with the so-called spatial change-of-support problem. The original version of **FRK**, which we refer to as **FRK** v.1, catered for Gaussian data only. This paper presents extensions to the package that allow it to cater for many distributions within the exponential family using a generalised linear modelling framework. Furthermore, the upgraded package, which we refer to as **FRK** v.2, allows for the use of many more basis functions when modelling the spatial process, and can therefore also often achieve more accurate predictions in a Gaussian setting than **FRK** v.1.

There are several approaches to spatial and spatio-temporal modelling in a non-Gaussian setting. One widespread method to deal with non-Gaussian data is *trans-Gaussian kriging* (Cressie 1993, pg. 137–138), in which standard kriging (i.e., spatial optimal linear prediction) is used after applying a non-linear transformation to the data. Several other approaches hinge on the use of a spatial version of the generalised linear mixed model (GLMM), whereby the response distribution is assumed to be a member of the exponential family, and the conditional mean is modelled using a transformation of some latent spatial process $Y(\cdot)$. The spatial GLMM was pioneered by Diggle, Tawn, and Moyeed (1998), who used a stationary model for $Y(\cdot)$ and a Markov chain Monte Carlo (MCMC) algorithm to obtain posterior predictive distributions. These methods typically suffer from computational inefficiencies in a big data setting, and some form of dimension reduction is required. Lopes, Gamerman, and Salazar (2011) used the spatial GLMM framework, but with a reduced-rank factor analytic model for $Y(\cdot)$. Lindgren and Rue (2011) provided a low-rank method using stochastic partial differential equations to link Gaussian fields to Gaussian Markov random fields, combining the interpretability of the former with the computational efficiency of the latter. Sengupta and Cressie (2013) employed the spatial GLMM framework, and modelled the latent process $Y(\cdot)$ as a linear combination of a fixed number of spatial basis functions with random coefficients (Cressie and Johannesson 2008). The basis-function coefficients were modelled

using an unconstrained covariance matrix, and Laplace approximations in an expectation maximisation algorithm resulted in maximum likelihood estimates of the unknown parameters. Finally, the empirical predictive distribution was generated using an MCMC algorithm. Lee and Park (2020) used a clustering algorithm to partition the spatial domain into disjoint subregions, and then for each subregion, a spatial GLMM model with a thin-plate-splines representation for $Y(\cdot)$, was used independently of the other subregions. The global process was then constructed as a weighted sum of the local processes. They modelled the basis-function coefficients as being independent.

Despite the plethora of modelling approaches available, software for spatial and spatio-temporal model fitting with non-Gaussian data is relatively scarce; see, for instance, the CRAN Task Views 'Analysis of Spatial Data' (Bivand 2020) and 'Handling and Analyzing Spatio-Temporal Data' (Pebesma 2020). A popular general purpose R package is **INLA** (Lindgren and Rue 2015), which uses the integrated nested Laplace Approximation (Rue, Martino, and Chopin 2009) approach for model fitting and prediction, and caters for a number of non-Gaussian distributions. In principle, it is capable of tackling the difficult modelling challenges that come with big, non-Gaussian, areal spatial and spatio-temporal data, however, as it is not intended for this purpose, it can be difficult for a user to implement. A project aimed at facilitating spatial modelling using **INLA** is the **inlabru** package (Bachl, Lindgren, Borchers, and Illian 2019), although spatio-temporal modelling was still not implemented at the time of writing. Generalised additive models (GAMs), which rely on constructing smooth functions of the covariates (in a spatio-temporal setting, the covariates would include space and time) can be implemented efficiently with the package **mgcv** (Wood 2017). The package yields excellent computational times, however the package does not cater for areal data or spatial change of support. The function `spGLM()` from the package **spBayes** (Finley, Banerjee, and Gelfand 2015) fits univariate Bayesian generalised linear spatial regression models. However, it only supports the modelling of binary and count (Poisson distributed) data in a non-Gaussian setting; it only caters for point-referenced data; and, as the basis functions depend on covariance-function parameters, only a small number of predictive process knots can be used, which causes a high degree of smoothing. The package **spNNGP** (Finley, Datta, and

Banerjee 2020) facilitates dimension reduction using a nearest neighbour Gaussian Process (Datta, Banerjee, Finley, and Gelfand 2016), and conducts inference using an efficient data augmented Gibbs sampler. It currently caters only for point-referenced spatial data, and is limited to binomial data in a non-Gaussian setting.

The main purpose of this article is to describe developments in **FRK** v.2 that allow for the modelling of 'big', non-Gaussian spatial and spatio-temporal data in a user-friendly and computationally efficient manner. In addition, **FRK** v.2 can now use significantly more basis functions than was possible in **FRK** v.1. The package can handle both point- and areally-referenced data, and deals with the spatial change-of-support problem. To our knowledge, this is the first time that a flexible GLMM framework built on a fixed rank spatial random effects model, with the capacity to include observations with differing supports, has been made accessible in a user-friendly package.

The remainder of this paper is organised as follows. In Section 2, we establish the proposed statistical framework, and describe model fitting and prediction using the R package **TMB** (Kristensen, Nielsen, Berg, Skaug, and Bell 2016). In Section 3, we discuss the new functionality in **FRK** v.2, provide illustrative examples using simulated data, and demonstrate how using an increased number of basis functions affects predictive performance. In Section 4, we present a comparative study between **FRK** v.2 and several related packages, as well as examples of real applications in which **FRK** v.2 may be useful. Section 5 concludes.

2. Methodology

The statistical model used in **FRK** v.2 is a spatial GLMM (Diggle *et al.* 1998), a hierarchical model consisting of two layers. In the *process layer*, we model the conditional mean of the data as a transformation of a latent spatial process, where the spatial process is modelled as a low-rank spatial random effects model (Cressie and Johannesson 2008; Sengupta and Cressie 2013); see Section 2.1. In the *data layer*, we use a conditionally independent exponential-family model for the data; see Section 2.2. In Section 2.3 we discuss parameter estimation, and in Section 2.4 we discuss spatial prediction and uncertainty quantification of predictions.

In Section 2.5 we outline our approach for spatio-temporal data.

2.1. The process layer

The process layer, which governs the conditional mean of the data, retains many similarities to that in **FRK** v.1, as described by Zammit-Mangion and Cressie (2021). Note that the following considers the spatial case only; the extension to a spatio-temporal setting is given in Section 2.5.

Denote the latent spatial process as $Y(\cdot) \equiv \{Y(\boldsymbol{s})\colon \boldsymbol{s} \in D\}$, where $\boldsymbol{s}$ indexes space in the spatial domain of interest D. The model for the latent process is

$$Y(\boldsymbol{s}) = \boldsymbol{t}(\boldsymbol{s})^\top \boldsymbol{\alpha} + v(\boldsymbol{s}) + \xi(\boldsymbol{s}); \quad \boldsymbol{s} \in D, \tag{1}$$

Each term in (1) is intended to capture a different form of spatial variability. First, spatially referenced covariates $\boldsymbol{t}(\cdot)$, and the associated regression parameters $\boldsymbol{\alpha}$, capture spatial variability that is linked to known, usually large-scale, explanatory variables. The model requires that the covariates are known at every location in D. Second, the spatially correlated random effect $v(\boldsymbol{s})$ captures medium-to-small scale spatial variation. Accounting for only large and medium-to-small scale spatial variation can result in overly smooth prediction surfaces, which in turn may lead to overly optimistic predictions; this problem is alleviated by including a fine-scale random process, $\xi(\cdot)$, which is 'almost' uncorrelated.

The medium-to-small scale spatial variation term $v(\cdot)$ is constructed as a linear combination of r spatial basis functions and random effects. Specifically,

$$v(\boldsymbol{s}) = \sum_{l=1}^{r} \phi_l(\boldsymbol{s})\eta_l = \boldsymbol{\phi}(\boldsymbol{s})^\top \boldsymbol{\eta}; \quad \boldsymbol{s} \in D,$$

where $\boldsymbol{\eta} \equiv (\eta_1, \ldots, \eta_r)^\top$ is an r-variate random effect vector and $\boldsymbol{\phi}(\boldsymbol{s}) \equiv (\phi_1(\boldsymbol{s}), \ldots, \phi_r(\boldsymbol{s}))^\top$ is an r-dimensional vector of pre-specified spatial basis functions evaluated at location $\boldsymbol{s}$. See Zammit-Mangion and Cressie (2021) for details on how these basis functions are constructed.

FRK v.1/v.2 discretises the domain of interest D into N small, non-overlapping basic areal

units (BAUs) $\{A_i : i = 1, \ldots, N\}$ such that $D = \cup_{i=1}^{N} A_i$. BAUs are a key element of **FRK** v.1/v.2, as they provide a framework that allows one to use both point-referenced and areal data simultaneously, and one that facilitates the spatial change-of-support problem. After discretisation via the BAUs, we obtain the vectorised version of (1),

$$\boldsymbol{Y} = \boldsymbol{T}\boldsymbol{\alpha} + \boldsymbol{S}\boldsymbol{\eta} + \boldsymbol{\xi}, \tag{2}$$

where $\boldsymbol{Y}$ is an N-dimensional vector, $\boldsymbol{T}$ and $\boldsymbol{S}$ are known design matrices, and $\boldsymbol{\xi}$ is the vector associated with the fine-scale process.

As in **FRK** v.1, we model the fine-scale random effects as being independent and identically distributed Gaussian random variables with variance σ^2_ξ, and we model $\boldsymbol{\eta}$ as a mean-zero multivariate-Gaussian random variable. **FRK** v.2 allows parameterisation of $\boldsymbol{\eta}$ in terms of a prior covariance matrix, $\boldsymbol{K}$, or in terms of a prior precision matrix, $\boldsymbol{Q}$. Both formulations use block-diagonal matrices, wherein basis-function coefficients are independent of the basis-function coefficients in differing resolutions. See Appendix A for details on the intra-resolution dependencies. For computational reasons, in a non-Gaussian setting (i.e., when **TMB** is used for model fitting) we typically recommend using $\boldsymbol{Q}$.

Following standard generalised linear model theory (McCullagh and Nelder 1989), **FRK** v.2 uses a link function, $g(\cdot)$, to model $Y(\cdot)$ as a transformation of the mean process, $\mu(\cdot)$:

$$g\left(\mu(\boldsymbol{s})\right) = Y(\boldsymbol{s}); \quad \boldsymbol{s} \in D.$$

The mean process evaluated over the BAUs is

$$\mu_i = g^{-1}(Y_i),\ i = 1, \ldots, N,$$

where $g^{-1}(\cdot)$ is the inverse link function. **FRK** v.2 is thus backward compatible: An identity link function and a Gaussian data model yields the model used in **FRK** v.1.

2.2. The data layer

Given m observations with footprints spanning one or more BAUs, we define the observation supports as $B_j \equiv \cup_{i \in c_j} A_i$ for $j = 1, \ldots, m$, where c_j is a non-empty set in the power set of $\{1, \ldots, N\}$, and define $D^O \equiv \{B_j : j = 1, \ldots, m\}$. Let $Z_j \equiv Z(B_j)$, $j = 1, \ldots, m$. The vector of observations (the data vector) is then $\boldsymbol{Z} \equiv (Z_1, \ldots, Z_m)^\top$.

Since each $B_j \in D^O$ is either a BAU or a union of BAUs, one can construct an $m \times N$ matrix

$$\boldsymbol{C}_Z \equiv \Big(w_i \mathbb{I}(A_i \subset B_j) : i = 1, \ldots, N; j = 1, \ldots, m \Big),$$

where $\mathbb{I}(\cdot)$ is the indicator function, which creates a linear mapping from $\boldsymbol{\mu} \equiv (\mu_i : i = 1, \ldots, N)^\top$ to evaluations of the mean process over the observation supports;

$$\boldsymbol{\mu}_Z \equiv \boldsymbol{C}_Z \boldsymbol{\mu}. \tag{3}$$

In **FRK** v.2, the weights w_i may be controlled through the `wts` field of the BAU object. If `wts` is `NULL`, each w_i is set to one, so that all BAUs are equally weighted. The `normalise_wts` argument in `SRE()` controls whether the linear mapping of $\boldsymbol{C}_Z$ corresponds to a weighted sum or a weighted average; if `normalise_wts = TRUE` (default, and implicit in **FRK** v.1), then the weights w_i are normalised so that the rows of $\boldsymbol{C}_Z$ sum to one, and the mapping represents a weighted average.

We assume that $[Z_j \mid \mu(\cdot), \psi] = [Z_j \mid \boldsymbol{\mu}_{Z,j}, \psi]$, where ψ is a dispersion parameter discussed below, and, for a generic random quantities A and B, $[A \mid B]$ denotes the probability distribution of A given B. That is, a given observation depends only on the value of the mean process at the corresponding observation support, rather than on the process over the whole domain. As a result, conditional on the latent spatial process, all observations are conditionally independent:

$$[\boldsymbol{Z} \mid \mu(\cdot), \psi] = \prod_{j=1}^{m} [Z_j \mid \boldsymbol{\mu}_{Z,j}, \psi].$$

We model the conditional distribution $[Z_j \mid \boldsymbol{\mu}_{Z,j}, \psi]$ as a member of the exponential family (McCullagh and Nelder 1989, Sec. 2.2.2), with conditional expectation $\mu(B_j) \equiv \mathbb{E}\left\{ Z_j \mid \boldsymbol{\mu}_{Z,j}, \psi \right\}$. Members of the exponential family may also be associated with a dispersion

parameter, ψ, which we assume is spatially invariant. Note that $\psi = 1$ for some distributions, (e.g., the binomial, negative-binomial, and Poisson distributions).

The model employed by **FRK** v.2 can be summarised as follows.

$$Z_j \mid \boldsymbol{\mu}_{Z,j}, \psi \overset{\text{ind}}{\sim} \text{EF}(\boldsymbol{\mu}_{Z,j}, \psi), \quad j = 1, \ldots, m, \tag{4}$$

$$\boldsymbol{\mu}_Z = \boldsymbol{C}_Z \boldsymbol{\mu}, \tag{5}$$

$$g(\boldsymbol{\mu}) = \boldsymbol{Y}, \tag{6}$$

$$\boldsymbol{Y} = \boldsymbol{T}\boldsymbol{\alpha} + \boldsymbol{S}\boldsymbol{\eta} + \boldsymbol{\xi}, \tag{7}$$

$$\boldsymbol{\eta} \mid \boldsymbol{\vartheta} \sim \text{Gau}(\boldsymbol{0}, \boldsymbol{Q}^{-1}), \tag{8}$$

$$\boldsymbol{\xi} \mid \sigma^2_\xi \sim \text{Gau}(\boldsymbol{0}, \sigma^2_\xi \boldsymbol{V}). \tag{9}$$

where $\boldsymbol{V}$ is a known diagonal matrix with positive entries on the diagonal and σ^2_ξ is either unknown and estimated, or provided by the user. In a spatio-temporal setting, a more complex model for $\boldsymbol{\xi}$ is allowed; see Section 2.5.

2.3. Estimation

We now derive the likelihood functions required for model fitting, describe how we deal with the intractable integrals which arise when using non-Gaussian data models, and describe how **TMB** (Kristensen *et al.* 2016) is used to obtain estimates of the parameters and fixed effects, and predictions of the random effects.

The complete-data likelihood function for our model is

$$L(\boldsymbol{\theta}; \boldsymbol{Z}, \boldsymbol{\eta}, \boldsymbol{\xi}) \equiv [\boldsymbol{Z}, \boldsymbol{\eta}, \boldsymbol{\xi}] = [\boldsymbol{Z} \mid \boldsymbol{\mu}_Z, \psi][\boldsymbol{\eta} \mid \boldsymbol{\vartheta}][\boldsymbol{\xi} \mid \sigma^2_\xi], \tag{10}$$

where $\boldsymbol{\theta} \equiv (\boldsymbol{\alpha}^\top, \boldsymbol{\vartheta}^\top, \sigma^2_\xi, \psi)^\top$ and $\boldsymbol{\vartheta}$ denotes the variance components associated with either $\boldsymbol{K}$ or $\boldsymbol{Q}$. The complete-data log-likelihood function is simply the logarithm of (10),

$$l(\boldsymbol{\theta}; \boldsymbol{Z}, \boldsymbol{\eta}, \boldsymbol{\xi}) \equiv \ln L(\boldsymbol{\theta}; \boldsymbol{Z}, \boldsymbol{\eta}, \boldsymbol{\xi}) = \ln [\boldsymbol{Z} \mid \boldsymbol{\mu}_Z, \psi] + \ln [\boldsymbol{\eta} \mid \boldsymbol{\vartheta}] + \ln [\boldsymbol{\xi} \mid \sigma^2_\xi]. \tag{11}$$

Under our modelling assumptions, the conditional density functions $[\boldsymbol{\eta} \mid \boldsymbol{\vartheta}]$ and $[\boldsymbol{\xi} \mid \sigma_\xi^2]$ are invariant to the specified link function and assumed distribution of the response variable. Of course, this invariance does not hold for $[\boldsymbol{Z} \mid \boldsymbol{\mu}_Z, \psi]$. As we only consider data models in the exponential family, $\ln[\boldsymbol{Z} \mid \boldsymbol{\mu}_Z, \psi]$ may be expressed as

$$\ln[\boldsymbol{Z} \mid \boldsymbol{\mu}_Z, \psi] = \sum_{j=1}^{m} \left\{ \frac{Z_j \lambda(\mu_{Z,j}) - b(\lambda(\mu_{Z,j}))}{a(\psi)} + c(Z_j, \psi) \right\}, \tag{12}$$

where $a(\cdot)$, $b(\cdot)$, and $c(\cdot, \cdot)$ are deterministic functions specific to the exponential family member, and $\lambda(\cdot)$ is the canonical parameter.

The marginal likelihood, which does not depend on the random effects, is given by

$$L^*(\boldsymbol{\theta}; \boldsymbol{Z}) \equiv \int_{\mathbb{R}^p} L(\boldsymbol{\theta}; \boldsymbol{Z}, \boldsymbol{u}) \mathrm{d}\boldsymbol{u}, \tag{13}$$

where $\boldsymbol{u} \equiv (\boldsymbol{\eta}^\top, \boldsymbol{\xi}^\top)^\top \in \mathbb{R}^p$, and p is the total number of random effects in the model. When the data is non-Gaussian, the integral in (13) is typically intractable and must be approximated either numerically or analytically. In **FRK** v.2, we use the Laplace approximation.

Let $\hat{\boldsymbol{u}} \equiv \hat{\boldsymbol{u}}(\boldsymbol{\theta}, \boldsymbol{Z})$ be a mode of $l(\boldsymbol{\theta}; \boldsymbol{Z}, \boldsymbol{u})$ with respect to $\boldsymbol{u}$, and let

$$\boldsymbol{H} \equiv - \left(\nabla_{\boldsymbol{u}} \nabla_{\boldsymbol{u}} l(\boldsymbol{\theta}; \boldsymbol{Z}, \boldsymbol{u}) \big|_{\boldsymbol{u}=\hat{\boldsymbol{u}}} \right)^{-1},$$

where $\nabla_{\boldsymbol{u}}$ denotes the gradient with respect to $\boldsymbol{u}$. A second-order Taylor series approximation of $l(\boldsymbol{\theta}; \boldsymbol{Z}, \boldsymbol{u})$ about $\boldsymbol{u} = \hat{\boldsymbol{u}}$ results in an approximation of (10) that is Gaussian in terms of $\boldsymbol{u}$, with mean vector $\hat{\boldsymbol{u}}$ and covariance matrix $\boldsymbol{H}$. Substituting this approximation into (13) and evaluating the integral yields the Laplace approximation of the marginal likelihood, $L^*(\boldsymbol{\theta}; \boldsymbol{Z}) \approx L(\boldsymbol{\theta}; \boldsymbol{Z}, \hat{\boldsymbol{u}})(2\pi)^{\frac{p}{2}} |\boldsymbol{H}|^{\frac{1}{2}}$.

Note that $[\boldsymbol{u} \mid \boldsymbol{Z}, \boldsymbol{\theta}] \propto [\boldsymbol{u}, \boldsymbol{Z} \mid \boldsymbol{\theta}]$, and $[\boldsymbol{u}, \boldsymbol{Z} \mid \boldsymbol{\theta}]$ is equivalent to the complete-data likelihood function, $L(\boldsymbol{\theta}; \boldsymbol{Z}, \boldsymbol{u})$. Since the Laplace approximation replaces $L(\boldsymbol{\theta}; \boldsymbol{Z}, \boldsymbol{u})$ with a term that has the form of a $\mathrm{Gau}(\hat{\boldsymbol{u}}, \boldsymbol{H})$ distribution in terms of $\boldsymbol{u}$, it follows that, approximately, $\boldsymbol{u} \mid \boldsymbol{Z}, \boldsymbol{\theta} \sim \mathrm{Gau}(\hat{\boldsymbol{u}}, \boldsymbol{H})$. This makes prediction of $\boldsymbol{u}$ and nonlinear functions of $\boldsymbol{u}$ via Monte Carlo simulation straightforward; see Section 2.4.

Model fitting with ***TMB***

Given as input a C++ template function which defines the complete-data log-likelihood function (11), **TMB** (Kristensen *et al.* 2016) computes the Laplace approximation of the marginal log-likelihood, and automatically computes its derivatives, which are then called from within **FRK** v.2 by an optimising function specified by the user (`nlminb()` is used by default). **TMB** uses **CppAD** (Bell 2005) for automatic differentiation, and the linear algebra libraries **Eigen** (Guennebaud, Jacob *et al.* 2010) and **Matrix** (Bates, Maechler, and Davis 2019) for vector and matrix operations in C++ and R, respectively; use of these packages yields good computational efficiency. **TMB**'s implementation of automatic differentiation is a key reason why we can cater for a variety of response distributions and link functions, as we do not need to consider each combination on a case-by-case basis.

Note that all parameters, fixed effects, and random effects, are treated as random in **TMB** (with a flat prior assumed if a prior is not provided). We fix the parameters to the estimates of their posterior modes and then treat them as non-random quantities when doing prediction; however, we let the user decide if the fixed effects $\boldsymbol{\alpha}$ are treated as fixed or random. If they fixed, flagged by setting `kriging = "simple"` in the function `predict()`, they are treated in the same way as the parameters. If they are random, we jointly sample $\boldsymbol{\alpha}$ along with the random effects $\boldsymbol{u}$ in the prediction stage; this choice accounts for the uncertainty in estimation of $\boldsymbol{\alpha}$, and so it is flagged by setting `kriging = "universal"` (see reference (19??) for the equivalence of using a flat prior on $\boldsymbol{\alpha}$ to performing universal kriging).

2.4. Prediction and uncertainty quantification

We now discuss prediction, and quantifying the uncertainty of those predictions. There are three primary quantities of interest in this framework: The latent process $Y(\cdot)$, the mean process $\mu(\cdot)$, and the noisy data process. To produce predictions and associated uncertainties, we need to determine the posterior distribution of these quantities.

Recall from Section 2.3 that the Laplace approximation implies that $\boldsymbol{u} \mid \boldsymbol{Z}, \boldsymbol{\theta}$ is approximated to be Gaussian. This, in turn, implies that the posterior distribution of $\boldsymbol{Y}$ is also approximated

to be Gaussian, and hence inference on $Y(\cdot)$ can be done using closed form solutions. However, the posterior distribution of non-linear functions of $Y(\cdot)$ (e.g., the mean process) are typically not available in closed form, and in this case some form of approximation is required. Hence, we choose to use a Monte Carlo framework.

Let $\boldsymbol{Y}_{\text{MC}}$ be an $N \times n_{\text{MC}}$ matrix whose columns are Monte Carlo samples of $\boldsymbol{Y} \mid \boldsymbol{Z}, \boldsymbol{\theta}$. Recall that $\boldsymbol{Y} = \boldsymbol{T}\boldsymbol{\alpha} + \boldsymbol{S}\boldsymbol{\eta} + \boldsymbol{\xi}$. If `kriging = "simple"`, then $\boldsymbol{u} = (\boldsymbol{\eta}^\top, \boldsymbol{\xi}^\top)^\top$ and $\boldsymbol{Y} = \boldsymbol{T}\boldsymbol{\alpha} + [\boldsymbol{S}\ \boldsymbol{I}]\,\boldsymbol{u}$, so we construct $\boldsymbol{Y}_{\text{MC}}$ via

$$\boldsymbol{Y}_{\text{MC}} \equiv \boldsymbol{T}\boldsymbol{A} + [\boldsymbol{S}\ \boldsymbol{I}]\,\boldsymbol{U}, \tag{14}$$

where $\boldsymbol{A}$ is a matrix with n_{MC} columns each of which contains the estimated posterior mode of $\boldsymbol{\alpha}$, and $\boldsymbol{U}$ is a matrix with n_{MC} columns consisting of Monte Carlo samples of $\boldsymbol{u} \mid \boldsymbol{Z}, \boldsymbol{\theta}$. If `kriging = "universal"`, then $\boldsymbol{\alpha}$ are treated as random effects and included in $\boldsymbol{u}$, and (14) is rewritten as

$$\boldsymbol{Y}_{\text{MC}} \equiv [\boldsymbol{T}\ \boldsymbol{S}\ \boldsymbol{I}]\,\boldsymbol{U}, \tag{15}$$

where $\boldsymbol{U}$ now also includes samples of $\boldsymbol{\alpha} \mid \boldsymbol{Z}, \boldsymbol{\theta}$. Note that we obtain estimates of the posterior mode and precision matrix of the random effects using **TMB**. We obtain Monte Carlo samples of $\boldsymbol{\mu}$ via $\boldsymbol{M} \equiv g^{-1}(\boldsymbol{Y}_{\text{MC}})$, where $g^{-1}(\cdot)$ is applied element-wise. Finally, Monte Carlo samples of the noisy data process can be constructed straightforwardly using $\boldsymbol{M}$, since the distribution of an exponential family member depends only on its conditional mean (and a dispersion parameter, which we model as being constant throughout the spatial domain).

For each quantity, we use the posterior expectation as our predictor. Predictions of $\boldsymbol{Y}$, $\boldsymbol{\mu}$, and the noisy data process over the BAUs can be computed by simply taking row-wise averages of the matrices defined above.

A commonly used metric for uncertainty quantification is the root-mean-squared prediction error (RMSPE). In a non-Gaussian setting, it can be difficult to interpret the RMSPE, and it is often more intuitive to quantify uncertainty through the width of the posterior predictive intervals. Hence, in **FRK** v.2, we also provide the user with user-specified percentiles of the posterior predictive distribution. These quantities can be computed straightforwardly using the Monte Carlo sampling approach described above.

Arbitrary prediction regions

Often, one does not wish to predict over a single BAU, but over regions spanning multiple BAUs. Define the set of prediction regions as $D^P \equiv \{\tilde{B}_k : k = 1, \ldots, N_P\}$, where $\tilde{B}_k \equiv \cup_{i \in c_k} A_i$, and where c_k is some non-empty set in the power set of $\{1, \ldots, N\}$. Like the data, the prediction regions $\{\tilde{B}_k\}$ may overlap. In practice, $\tilde{B}_k$ may not include entire BAUs; in this case, we assume that a prediction region contains a BAU if and only if there is at least some overlap between the BAU and the prediction region. **FRK** v.1 assumed that a prediction region contains a BAU if and only if the BAU centroid lies within the region; the relaxed condition in **FRK** v.2 is intended to cater for non-convex BAUs, such as those used in Section 4.3. Prediction over D^P requires some form of aggregation across relevant BAUs. Since in the non-Gaussian setting aggregation must be done on the original scale, we restrict prediction over arbitrary regions to the mean (or the noisy data process). Therefore, predictions of the latent process $Y(\cdot)$ are not allowed over arbitrary prediction regions.

Consider the predictions $\{\mu_P(\tilde{B}_k) : k = 1, \ldots, N_P\}$, where $\mu_P(\cdot) \equiv \mu(\cdot \mid \boldsymbol{Z}, \boldsymbol{\theta})$. These predictions are weighted sums of the predictions over the associated BAUs. Specifically,

$$\mu_{P,k} \equiv \mu_P(\tilde{B}_k) = \sum_{i=1}^{N} \tilde{w}_i \mathbb{I}(A_i \subset \tilde{B}_k)\mu_i; \quad i = 1, \ldots, N;\ k = 1, \ldots, N_P;\ \tilde{B}_k \in D^P,$$

where, in a similar fashion to the incidence matrix $\boldsymbol{C}_Z$, the weights $\{\tilde{w}_i\}$ are optionally provided by the user in the `wts` field of the BAU object, and may be normalised if `normalise_wts = TRUE`. If `wts` is `NULL`, the BAUs are assumed to be equally weighted. Define $\boldsymbol{\mu}_P \equiv (\mu_{P,k} : k = 1, \ldots, N_P)^\top$. Since each element in D^P is the union of subsets of D^G, one can construct a matrix,

$$\boldsymbol{C}_P \equiv (\tilde{w}_i : i = 1, \ldots, N; k = 1, \ldots, N_P)\,,$$

such that

$$\boldsymbol{\mu}_P = \boldsymbol{C}_P \boldsymbol{\mu}.$$

We use Monte Carlo sampling to predict $\boldsymbol{\mu}_P$. Monte Carlo samples of $\boldsymbol{\mu}_P \mid \boldsymbol{Z}, \boldsymbol{\theta}$ can be

constructed straightforwardly via $\boldsymbol{M}_P \equiv \boldsymbol{C}_P\boldsymbol{M}$, where $\boldsymbol{M}$ is defined above.

2.5. Spatio-temporal framework

Extending **FRK** v.2 to accommodate non-Gaussian spatio-temporal data involves using an additional dimension of basis functions. Let r_t and r_s denote the number of temporal and spatial basis functions, respectively. Denote $\boldsymbol{Q}_t$ and $\boldsymbol{Q}_s$ as the precision matrices of the random coefficients associated with the temporal basis functions and spatial basis functions, respectively. We model the $r_t r_s \times r_t r_s$ precision matrix of the $r_t r_s$ spatio-temporal random coefficients associated with basis functions created through the tensor product of the r_t temporal and r_s spatial basis functions as

$$\boldsymbol{Q} = \boldsymbol{Q}_t \otimes \boldsymbol{Q}_s.$$

The assumption of separability between time and space, and hence the ability to write $\boldsymbol{Q}$ as a Kronecker product, leads to significant computational savings. **FRK** v.2 uses an AR1 model for the random coefficients associated with the temporal basis functions.

In a spatio-temporal setting, it is possible that each spatial BAU is observed multiple times. Furthermore, it is also possible that the fine-scale variation is not constant over the spatial domain D. In these situations, a better fit may be obtained by allowing each spatial BAU to be associated with its own fine-scale variance parameter. Let N_s and N_t denote the number of spatial and temporal BAUs, respectively (so that $N = N_s N_t$). Then, instead of modelling $\boldsymbol{\xi} \sim \text{Gau}(\boldsymbol{0}, \sigma^2_\xi \boldsymbol{I})$, **FRK** v.2 also allows one to model $\boldsymbol{\xi} \sim \text{Gau}\left(\boldsymbol{0}, \boldsymbol{\Sigma}_\xi\right)$, where

$$\boldsymbol{\Sigma}_\xi \equiv \begin{pmatrix} \text{diag}(\boldsymbol{\sigma}^2_\xi) & & \\ & \ddots & \\ & & \text{diag}(\boldsymbol{\sigma}^2_\xi) \end{pmatrix}, \tag{16}$$

$\boldsymbol{\sigma}^2_\xi \equiv (\sigma^2_{\xi,1}, \ldots, \sigma^2_{\xi,N_s})^\top$, and the BAUs are assumed to be ordered such that space runs faster than time. This model for $\boldsymbol{\Sigma}_\xi$ is flagged by setting `fs_by_spatial_BAU = TRUE` in the `SRE()` function, and is particularly useful when the number of spatial BAUs (and hence variance parameters to estimate) is relatively low and we have observed each spatial BAU over many

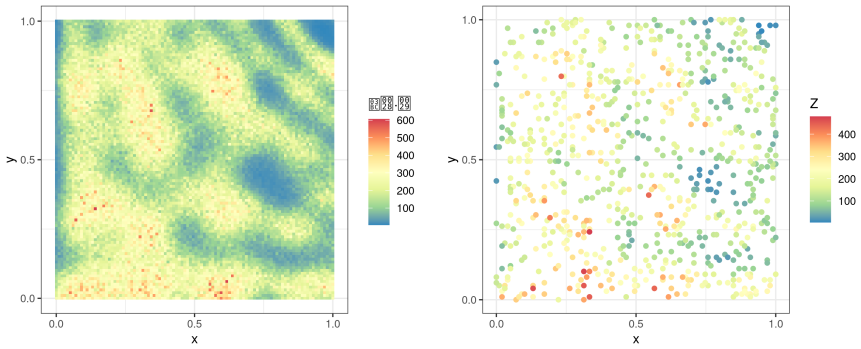

Figure 1: Simulated point-referenced spatial Poisson data set, and the true field from which the data was generated, used for the example presented in Section 3.1. (Left Panel) True mean process $\mu(\cdot)$. (Right panel) Simulated observations.

time-points; see, for instance, the study presented in Section 4.4.

3. Usage of new features

We now demonstrate the new features in **FRK** v.2, an overview of which is presented in Table 1. The primary new feature in **FRK** v.2 is the packages ability to cater for non-Gaussian data models: A full list of available data models and link functions is shown in Table 2. In Section 3.1, we illustrate use of **FRK** v.2 using non-Gaussian spatial point-referenced data. In Section 3.3, we briefly discuss extensions available in **FRK** v.2 to model data in a spatio-temporal setting. Finally, in Section 3.4, we show the potential improvement in predictive performance of **FRK** v.2 over **FRK** v.1, thanks to the support for an increased number of basis functions in **FRK** v.2.

3.1. Illustrative example: Simulated, non-Gaussian, point-referenced spatial data

For illustration, and so that readers can familiarise themselves with the workflow of **FRK** v.2, we use a Poisson data set containing 750 observations as a running example. The true mean process over D and the data, both shown in Figure 1, were simulated using code provided at `https://github.com/MattSainsbury-Dale/FRKv2_src`.

The first step when using **FRK** v.1/v.2 is to create basis functions and BAUs, a task that

Table 1: Important extensions to function arguments in **FRK** v.2.

Function	Argument	Use
`SRE()`	`response`	A string indicating the data model.
	`link`	A string indicating the link function.
	`K_type`	Controls the parameterisation of the prior variance matrix of the basis-function coefficients. If **TMB** is used for model fitting (which it must be for non-Gaussian data models), permissible values are `"block-exponential"` and `"precision"`, with the latter recommended for computational efficiency.
	`normalise_wts`	Flag controlling whether the linear mappings of $\boldsymbol{C}_Z$ and $\boldsymbol{C}_P$ correspond to a weighted sum or a weighted average; if `TRUE`, then the weights are normalised so that aggregation of the mean over BAUs is a weighted average.
	`fs_by_spatial_BAU`	Flag controlling whether each spatial BAU is given its own fine-scale variance parameter; only applicable in a spatio-temporal setting.
`SRE.fit()`	`method`	Controls the method of model fitting; the newly permissible value is `"TMB"`, which is required whenever a non-Gaussian data model or non-identity link function is used.
	`optimiser`	Optimising function when `method = "TMB"` (default `nlminb()`).
	`taper`	A positive scalar indicating the strength of the covariance tapering (see Appendix A).
	`known_sigma2fs`	Allows the user to fix the fine-scale variance to a known value. If `fs_by_spatial_BAU = TRUE`, the argument `known_sigma2fs` should be a vector of length equal to the number of spatial BAUs.
`predict()`	`newdata`	`newdata` can now be a `SpatialPoints*` or `STI*` object, facilitating prediction over spatial or spatio-temporal point-referenced locations.
	`type`	A vector of strings indicating the quantities of interest for which inference is desired. If `"link"` is in `type`, $Y(\cdot)$ is included; If `"mean"` is in `type`, the mean process (and the probability process, if applicable) is included; If `"response"` is in `type`, the response variable is included.
	`percentiles`	Numeric vector (each element between 0 and 100) indicating the percentiles of the posterior predictive distribution(s) that are to be computed.
	`kriging`	A character string indicating whether `"simple"` or `"universal"` kriging is to be performed.
	`n_MC`	The number of Monte Carlo samples at each BAU.
`auto_BAUs()`	`spatial_BAUs`	The spatial BAUs in a spatio-temporal setting. If `NULL`, the spatial BAUs are constructed automatically.
`plot()`	-	A method for plotting the data, predictions, and uncertainty quantification given the result of a call to `predict()`.

Table 2: Combinations of exponential family member response distributions and link functions available in **FRK** v.2. A '✓' indicates a combination is supported. A '•' indicates a combination is allowed, however, due to the implied range of μ, the support of the observations, and the form of probability density function of that family, nonsensical results are possible; if one of these problematic combinations is chosen, a warning is given to the user. Finally, blank entries indicate that the combination is not allowed.

		Link Function					
		identity	inverse	inverse-squared	log	square-root	logit/ probit/ cloglog
Family	Gaussian	✓	✓	•	•	•	
	Poisson	•	•	•	✓	✓	
	gamma	•	•	•	✓	✓	
	inverse-Gaussian	✓	✓	•	✓	✓	
	negative-binomial*				✓	✓	✓
	binomial*						✓

*The binomial and negative-binomial distributions have a known constant 'size' parameter, k_j, and a 'probability of success' parameter, π_j, associated with each datum, Z_j; see Appendix B for details on how we link the latent process $Y(\cdot)$ to the mean process $\mu(\cdot)$ when these distributions are chosen.

can be performed automatically using the helper functions `auto_BAUs()` and `auto_basis()`: See Zammit-Mangion and Cressie (2021) for details. Next, an '`SRE`' object is initialised using `SRE()`, within which we specify the data model, the link function, and the parameterisation of the basis-function coefficients. We then fit the model using `SRE.fit()`. These steps may be performed in a single line of code with the convenient wrapper function `FRK()`. Note that when the response is non-Gaussian or a non-identity link function is chosen, `FRK()` automatically selects `method = "TMB"` and `K_type = "precision"`.

```
R> S <- FRK(f = Z ~ 1, data = list(Poisson_simulated),
+    response = "poisson", link = "log")
```

Prediction is done using `predict()`. The argument `type`, a vector of strings, specifies the quantities of interest for which predictions and prediction uncertainty is desired: See Table 1. In this example, we set `type = c("link", "mean")` to obtain predictions for the latent process $Y(\cdot)$ and the mean $\mu(\cdot)$. The `percentiles` argument allows the computation of

percentiles of the returned posterior predictive distributions, and hence posterior predictive intervals; the default of `percentiles` is `c(5, 95)`, so that the 5th and 95th percentiles of the posterior predictive distribution are computed.

```
R> pred <- predict(S, type = c("link", "mean"))
```

When `method = "TMB"`, the returned object is a list containing two elements. The first element is an object of the same class as `newdata` (if `newdata` is unspecified, prediction is done over the BAUs), and contains the predictions and associated uncertainty quantifications (the RMSPE and percentiles of the posterior predictive distribution) of each term specified in `type`. The second element is a list of matrices containing Monte Carlo samples of the quantities of interest at each prediction location.

Finally, we generate a list of '`ggplot`' (Wickham 2016) objects of the predictions and associated uncertainty using the function `plot()`.

```
R> plots <- plot(S, pred)
```

The '`ggplot`' objects can then easily be arranged in a grid using various dedicated packages; in this article, we used the `ggarrange()` from the package **ggpubr** (Kassambara 2020). Figure 2 shows prediction and prediction uncertainty quantification for both the latent process $Y(\cdot)$ and the mean process $\mu(\cdot)$. The prediction of the mean process seems reasonable given the data. The prediction uncertainty of the latent process is relatively flat, with increases in uncertainty coinciding with regions of data paucity; on the other hand, the prediction uncertainty of the mean process is roughly proportional to its prediction. The 'bullseye' points of low uncertainty, visible in both prediction uncertainties, correspond to the observed locations. The point-like nature of this reduction in uncertainty is due to our assumption that the fine-scale random effects, $\boldsymbol{\xi}$, are mutually independent at the BAU level: BAUs neighbouring an observed BAU do not borrow strength from the inferred fine-scale random effect at the observed BAU.

3.2. Non-Gaussian, areally-referenced spatial data and change of support

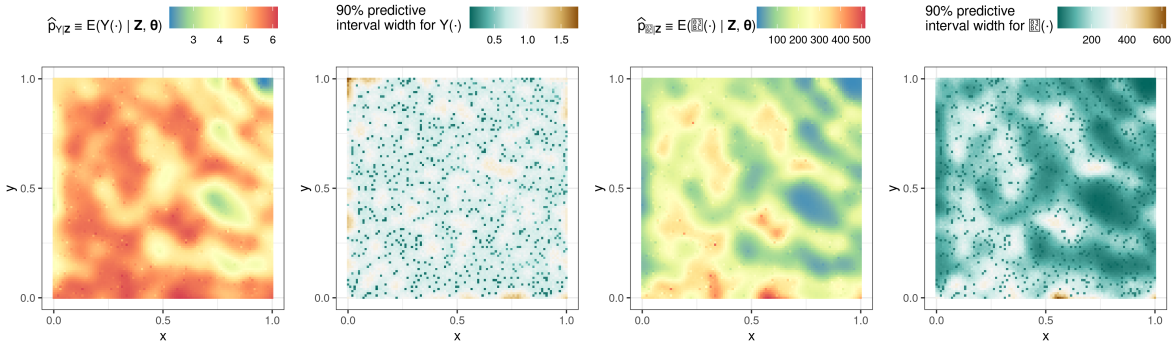

Figure 2: The prediction and prediction uncertainty quantification using the data shown in Figure 1. (Left panel) Prediction of the latent process, $Y(\cdot)$. (Centre-left panel) Width of the 90% central predictive interval of the latent process. (Centre-right panel) Prediction of the mean process, $\mu(\cdot)$. (Right panel) Width of the 90% central predictive interval of the mean process.

In this section, we illustrate **FRK** v.2 on non-Gaussian, *areally*-referenced, spatial data, as well as its use in predicting over large areas. We again provide a running example; this time, we analyse a simulated, areally-referenced negative-binomial data set.

We now describe data simulation, which is illustrated in Figure 3, and the code is provided at `https://github.com/MattSainsbury-Dale/FRKv2_src`. First, two square grids are defined: The first is at a fine resolution and corresponds to the BAUs, and the second is at a coarser resolution and corresponds to areal data supports. To get a handle on the fine-scale variation parameter, we use some of the BAUs as data supports: Hence we have a mixture of coarse- and fine-scale observations. We define the probability process evaluated over the BAUs by passing a sum of trigonometric functions through the logistic function. We then construct the mean process evaluated over the BAUs. Since we are simulating negative-binomial data, construction of the mean requires specification of a size parameter; for simplicity we use $k = 50$ for each BAU. We then aggregate the mean process at the BAU level over the data supports, and simulate data using the aggregated mean process. Finally, we exclude some observations to form a training set.

Now we construct and fit the '`SRE`' object using `FRK()`. When we have areal observations, or when we are predicting over polygons which comprise multiple BAUs, one should declare whether the mean process is to be averaged or summed (see Section 2.1). In many non-

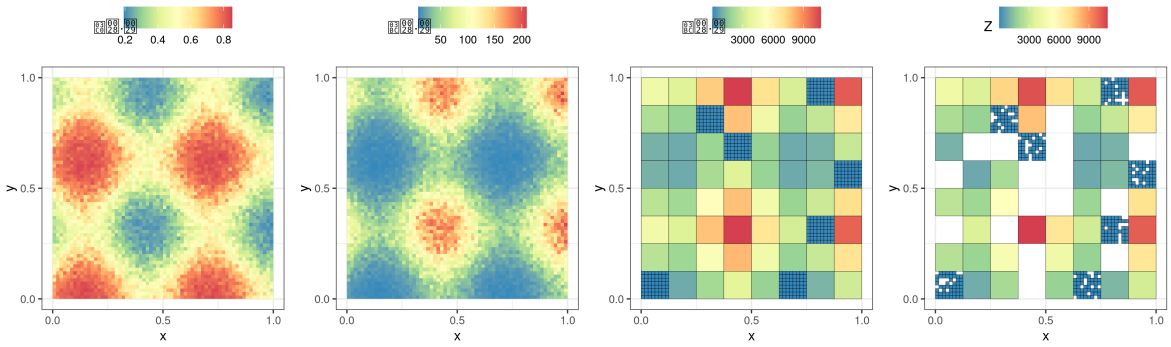

Figure 3: Simulated, areal negative-binomial data set used in the illustrative example of Section 3.2. (Left panel) True probability process evaluated over the BAUs. (Centre-left panel) True mean process evaluated over the BAUs. (Centre-right panel) True mean process aggregated to the data support level. (Right panel) Simulated data at the data support level, with some observations omitted; this is the data used for model fitting.

Gaussian applications, it is natural to sum the mean process over the BAUs; this instruction is given by setting `normalise_wts = FALSE`. Note that for binomial and negative-binomial data, the size parameter must be provided: In general, we need the size parameter of every observed BAU. When each observation is associated with exactly one BAU (e.g., point-referenced data), we allow users to provide the size parameter with the observations; when some observations are associated with multiple BAUs (e.g., areal data with large data supports), the user must provide the size parameter at the BAU level (for all observed BAUs).

```
R> S <- FRK(f = Z ~ 1, data = list(zdf), BAUs = BAUs,
+     response = "negative-binomial", link = "logit", normalise_wts = FALSE)
```

Next, we predict over the BAUs.

```
R> pred <- predict(S)
```

Figure 4 shows the prediction and prediction uncertainty quantification for both the mean process, $\mu(\cdot)$, and the probability process, $\pi(\cdot)$, at the BAU level: This graphic was constructed using `plot()`. We observe agreement between the fields shown in Figure 3 and the corresponding predictions. The prediction uncertainty of the mean process is roughly proportional to its prediction (as in the example of Section 3.1). In contrast, the prediction uncertainty of $\pi(\cdot)$ is low when the prediction is near 0 or 1, and increases when the prediction is near 0.5:

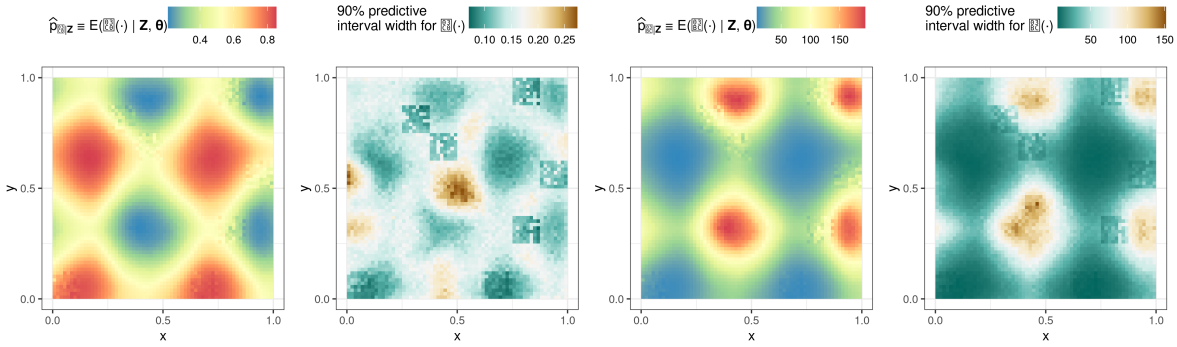

Figure 4: Prediction and prediction uncertainty quantification for the simulated negative-binomial areal toy example shown in Section 3.2. (Left panel) Prediction of the probability process $\pi(\cdot)$. (Centre-left panel) Width of the 90% central predictive interval of the probability process. (Centre-right panel) Prediction of the mean process, $\mu(\cdot)$. (Right panel) Width of the 90% central predictive interval of the mean process.

This is expected from properties of the negative-binomial distribution. Uncertainty in both quantities is lower over areas in which we have fine-scale data. The mean empirical coverage from the 90% posterior predictive intervals was 90.9%, which is almost nominal, and very reasonable given the difficulty inherent in doing spatial change of support.

To emphasise that the prediction polygons are unrelated to the BAUs and data supports, we demonstrate prediction over a handful of irregularly shaped areas (defined as a '`SpatialPolygons*`' object).

```
R> pred <- predict(S, newdata = arbitrary_polygons)
```

Recall that when we are predicting over arbitrary polygons, we restrict prediction to $\mu(\cdot)$. Figure 5 shows predictions for $\mu(\cdot)$ over polygons. This graphic was made using `plot()`.

3.3. Spatio-temporal extensions

FRK v.2 now also caters for non-Gaussian, point- and areally-referenced, *spatio-temporal* data. For the sake of brevity, we will not use a simple study to show this, and instead refer readers to the application study of Section 4.4.

3.4. Support for increased number of basis functions

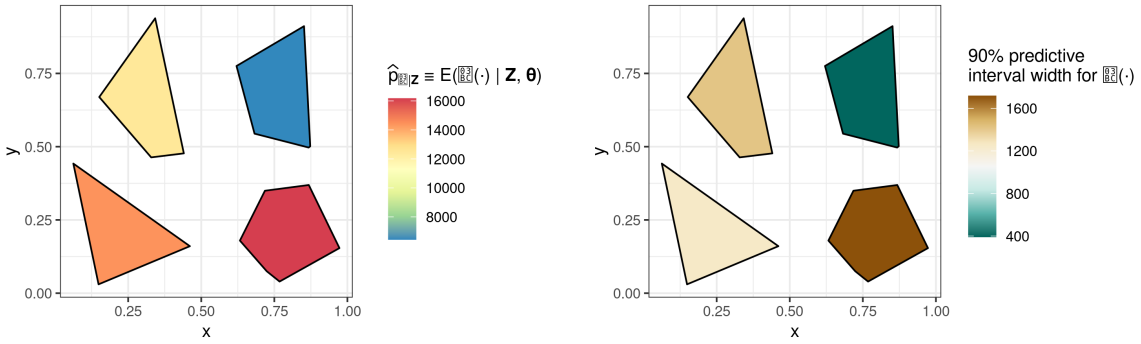

Figure 5: Prediction (left panel) and prediction uncertainty (right) of the mean process, $\mu(\cdot)$, when predicting over arbitrary polygons in the toy example of Section 3.2. Note that these polygons have equal area.

Table 3: Diagnostics comparing the predictive performance when using a range of basis function resolutions and with point-referenced count data. The diagnostics are the root mean-square prediction error (RMPSE), the continuous ranked probability score (CRPS), and the empirical coverage (Cvg90) and interval score (IS90) resulting from a central prediction interval with a nominal coverage of 90%. The diagnostics are in regards to prediction of the true mean process, $\mu(\cdot)$, and are averaged over all unobserved locations.

Resolutions (basis functions)	RMSPE	CRPS	Cvg90	IS90	Run Time (Min.)
1 (9)	83.15	47.63	0.896	377.26	0.067
2 (81)	53.96	28.54	0.893	224.24	0.128
3 (729)	47.12	24.31	0.895	182.59	0.521

The efficiency of **TMB** and our use of sparse precision matrices means that **FRK** v.2 is now better equipped than **FRK** v.1 to use a large number of basis functions. The predictive performance of the framework can be closely related to the number of basis functions, as shown in the following examples. Appendix C defines the scoring rules used throughout the remainder of this paper.

We repeated the analysis in Section 3.1 using one, two, and three resolutions of basis functions; Table 3 shows the results for each run. Clearly predictive performance improves as the number of basis functions increases. However, the coverage remains accurate in all runs, implying that the model is able to accurately quantify uncertainty irrespective of the number of basis functions. This important property is in large part due to the fine-scale random variation term, $\xi(\cdot)$, in (1).

Although the primary motivation for **FRK** v.2 is the provision of non-Gaussian data models,

Table 4: Numerical scoring for each competing method on the MODIS data, as presented in Heaton *et al.* (2019).

Method	MAE	RMSPE	CRPS	IS95	Cvg95	Run Time (Min.)	Cores Used
FRK v.2	1.30	1.69	0.92	8.32	0.93	72.27[a]	1[b]
FRK v.1	1.96	2.44	1.44	14.08	0.79	2.32	1
Gapfill	1.33	1.86	1.17	34.78	0.36	1.39	40
Lattice Krig	1.22	1.68	0.87	7.55	0.96	27.92	1
LAGP	1.65	2.08	1.17	10.81	0.83	2.27	40
Metakriging	2.08	2.50	1.44	10.77	0.89	2888.52	30
MRA	1.33	1.85	0.94	8.00	0.92	15.61	1
NNGP Conjugate	1.21	1.64	0.85	7.57	0.95	2.06	10
NNGP Response	1.24	1.68	0.87	7.50	0.94	42.85	10
Partition	1.41	1.80	1.02	10.49	0.86	79.98	55
Pred. Proc.	2.05	2.52	1.85	26.24	0.75	640.48	1
SPDE	1.10	1.53	0.83	8.85	0.97	120.33	2
Tapering	1.87	2.45	1.32	10.31	0.93	133.26	1
Periodic Embedding	1.29	1.79	0.91	7.44	0.93	9.81	1

[a] **FRK** v.2 was implemented in a different computing environment than the other models, and so run time is not directly comparable. **FRK** v.2 was implemented using a machine with 16 GB of RAM and an Intel i7-9700 3.00GHz CPU with 8 cores. The other models were implemented using the Becker computing environment (256 GB of RAM and 2 Intel Xeon E5-2680 v4 2.40GHz CPUs with 14 cores each and 2 threads per core - totaling 56 possible threads for use in parallel computing) located at Brigham Young University (Heaton *et al.* 2019).
[b] **TMB** supports the use of multiple cores, but this is not yet implemented in **FRK** v.2.

the ability to use a large number of basis functions also leads to higher-quality predictions in a Gaussian setting. We show this through the comparative study provided in Heaton *et al.* (2019). The data used in that study was made up of a training and a test set consisting of 105,569 and 42,740 observations, respectively; see Heaton *et al.* (2019) for a detailed description of the data. Table 4 replicates Table 3 of Heaton *et al.* (2019), with an additional entry corresponding to **FRK** v.2, wherein many more basis functions are used than was practical with **FRK** v.1. Specifically, **FRK** v.1 used 485 basis functions, whilst **FRK** v.2. uses 12114. The results show that the increased number of basis functions significantly improves the diagnostic scores of **FRK**, and the result are now comparable to those of MRA. To achieve these improvements over **FRK** v.1, we only had to specify `nres = 4` in the `auto_basis()` function, and then `K_type = "precision"` and `method = "TMB"` in the `SRE()` and `SRE.fit()` functions, respectively; the rest of the **FRK** code as used in the competition was left unchanged.

4. Application and comparison studies

We now provide several application case studies using **FRK** v.2. In Section 4.1, we present a comparison study between **FRK** v.2 and other packages which cater for non-Gaussian data models. In Section 4.2, we demonstrate block prediction using contaminated soil data, and compare our results to other modelling approaches. In Section 4.3, we use data on poverty figures in Sydney, Australia, to demonstrate the spatial change-of-support functionality of **FRK** v.2 in a non-Gaussian setting. In Section 4.4, we provide a non-Gaussian spatio-temporal example through modelling crime counts in the city of Chicago over the first two decades of the 21st century.

4.1. Comparative study: MODIS cloud data

In this section we compare out-of-sample predictions from **FRK** v.2 to those from the R packages **INLA** (Lindgren and Rue 2015), **spNNGP** (Finley *et al.* 2020), **spBayes** (Finley *et al.* 2015), and **mgcv** (Wood 2017) using a binary data set. The data form an image of a cloud taken by the Moderate Resolution Imaging Spectroradiometer (MODIS) instrument aboard the Aqua satellite (MODIS Characterization Support Team 2015). Data collected from the MODIS instrument have been used in several related works; see, for instance, Sengupta and Cressie (2016) and Zammit-Mangion, Ng, Vu, and Filippone (2021). For this comparative study, data pre-processing involved first coarsening the image from over 10 million pixels to a more manageable 33750 pixels, by creating a 150×225 grid and computing the mean value of the response within each grid cell. Then, as the data provided by the MODIS instrument is continuous (measuring radiance in units of $\mathrm{W/m^2/\mu m/st}$), we applied a reasonable threshold to obtain a binary version of the data (i.e., cloud, or no-cloud).

We considered two types of sampling schemes for model testing. The first was missing-at-random (MR), whereby we randomly selected a sub-sample of pixels to act as training data. Under the MR sampling scheme, we randomly sampled 6000 pixels for training, leaving 27750 pixels for testing. The second sampling scheme, which we refer to as 'missing-in-a-block' (MB), involved excluding all pixels within a block for training, and using pixels inside

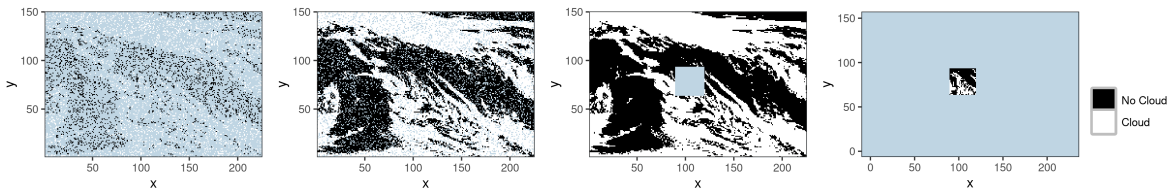

Figure 6: MODIS data used in the comparative study of Section 4.1; a blue background is used to make the 'No Cloud' and the 'Cloud' pixels easier to distinguish. (Left panel) The missing-at-random sample used for training. (Centre-left panel) The missing-at-random test data. (Centre-right panel) The 'missing-in-a-block' sample used for training. (Right panel) The 'missing-in-a-block' test data.

the block for testing. The block is a 30×30 square (900 pixels) in the middle of the spatial domain of interest. The training and test sets under the two sampling schemes are shown in Figure 6.

The software used in this study each required several modelling decisions, which had to be made in a way that balanced predictive performance and run time. We took a systematic approach to a pre-processing model-selection phase by splitting the training set in two, and then using one half for model fitting and the other half for model validation. In this way, we were able to test a large number of arguments for each package, and choose the best combination in terms of predictive performance and run time. For the methods requiring specification of a link function, we used the standard logit link function. For **FRK** v.2, we used four resolutions of basis functions; a total of 11,130 basis functions. For **INLA**, we discretised the domain into 8671 elements. We used the `bam()` function from **mgcv**, which is similar to generalised additive model function `gam()`, but optimised for large data sets, with 3000 knots. For **spBayes**, we used 400 knots; increasing the number of knots further was computationally prohibitive. We found that the default option of considering 15 neighbours at a time when using **spNNGP** was appropriate. The packages **spNNGP** and **spBayes** use Markov chain Monte Carlo (MCMC): At both training and test locations, we used 10000 total MCMC samples, a burn-in of 6000, and a thinning factor of 10; hence, 400 approximately independent samples from the predictive distribution of the process were available at each spatial location. The number of cores used for **spNNGP** can be controlled through the argument `n.omp.threads`. Setting `n.omp.threads` to be greater than 1 did not

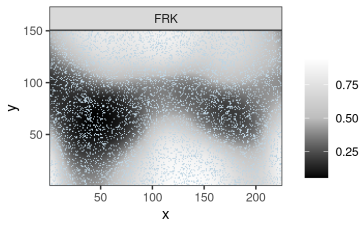

Figure 7: Predictions of the probability of cloud resulting from the missing-at-random data shown in Figure 6. Note that the training locations are indicated by blue pixels.

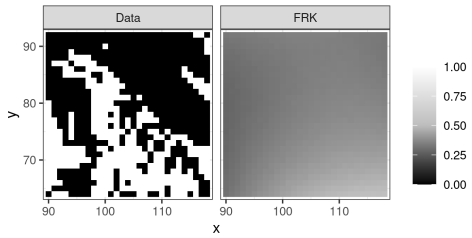

Figure 8: Predictions of the probability of cloud resulting from the 'missing-in-a-block' data shown in Figure 6. Here, we have shown only the testing locations, which corresponds to the 30×30 block near the centre of the spatial domain; the test data corresponding to this block is shown in the left-most panel of this figure.

work on our computing system (a known issue documented in the **spNNGP** package manual); hence, our reported run-times for **spNNGP** are for a single core.

For each method and each sampling scheme, we predicted the probability of cloud at each pixel. Figure 7 shows the predictions resulting from the MR data shown in Figure 6. The predictions of **FRK** v.2, **INLA**, and **spNNGP** are similar, while the predictions of **mgcv** are slightly smoother than those from the aforementioned packages. The predictions of **spBayes** are even smoother; this is due to the small number of knots. Figure 8 shows the predictions resulting from the MB data shown in Figure 6. **FRK** and **INLA** return predictive probabilities close to 0.5, while **mgcv** and **spBayes** are more confident in their predictions. There is an interesting pattern in the **spNNGP** predictions; this is an expected artefact of the nearest-neighbour approach.

The packages used in this study can provide prediction standard errors associated with the probability process. However, the underlying distribution of the probability process is unidentifiable, as the posterior predictive distribution, $Z^* \mid \boldsymbol{Z}$, for some validation datum Z^*, de-

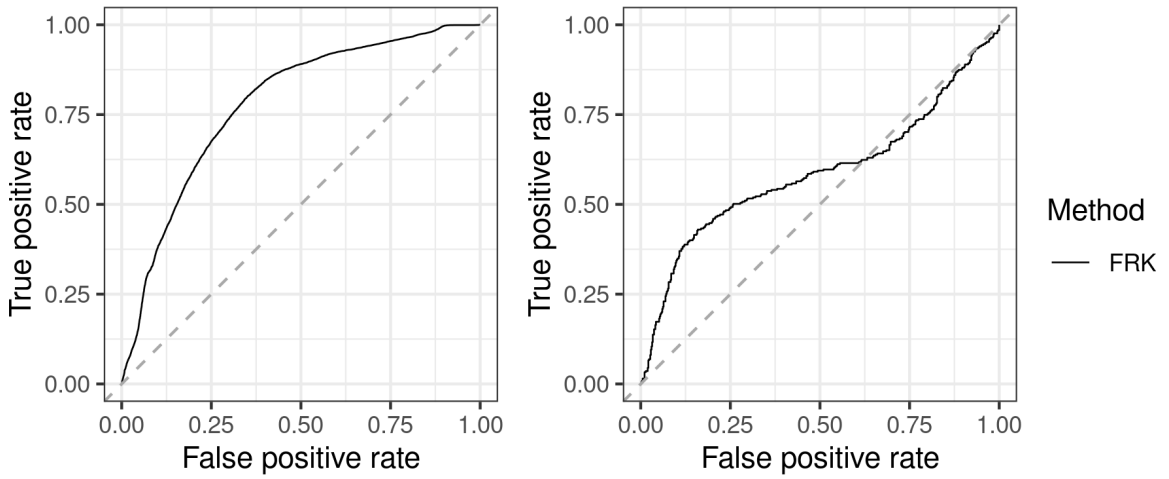

Figure 9: ROC curves for the training/test sets displayed in Figure 6. (Left panel) ROC curves generated from the 'missing-at-random' data. Note that there is some degree of overlap between **FRK**, **INLA**, and **spNNGP**. (Right panel) ROC curves generated from the 'missing-in-a-block' data.

pends only on the posterior expectation of the probability parameter at the corresponding location, $\mathbb{E}(\pi^* \mid \boldsymbol{Z})$. For this reason, we do not attempt to validate the prediction intervals, and instead focus our efforts on predictive accuracy.

To assess predictive accuracy, we compared the predictions from all models in terms of the Brier score (Gneiting, Balabdaoui, and Raftery 2007, Sec. 3), and the area under the receiver operating characteristic (ROC) curve (AUC). The Brier score assesses how close the predicted probability of cloud is to the truth; it is a negatively oriented rule, where low scores indicate

Table 5: Diagnostic results for the MODIS comparison study. Best performers for a given diagnostic are boldfaced.

Scheme	Method	Brier score	AUC	Run Time (Min.)
MR	FRK	0.087	0.955	**5.47**
	INLA	0.090	0.951	54.44
	mgcv	0.091	0.948	45.77
	spBayes	0.105	0.932	68.01
	spNNGP	**0.083**	**0.956**	12.35
MB	FRK	**0.192**	**0.769**	**9.29**
	INLA	0.200	0.757	141.85
	mgcv	0.219	0.701	186.67
	spBayes	0.247	0.632	489.41
	spNNGP	0.198	0.749	59.30

accurate predictions of the probability of cloud. In contrast, higher AUC scores are preferred. The results for each method and each sampling scheme are reported in Table 5; the ROC curves are shown in Figure 9. For the MR scheme, there is little discernible difference between **FRK**, **INLA**, **mgcv**, and **spNNGP**. However, as one may expect upon viewing the predictions in Figure 7, **spBayes** performs poorly in comparison to the other packages due to the small number of knots. The task of prediction over a completely unobserved region is challenging, and so it is no surprise that the diagnostics for the MB scheme are significantly worse than the MR scheme. In this case, we see **FRK**, **INLA**, and **spNNGP** performing slightly better than **mgcv**, which in turn performs better than **spBayes**. Note that all run times increased under the MB scheme; however, **FRK** v.2 increased by the smallest amount (increasing by a factor of less than 2), while the run times for other packages increased by between a factor of 3 and 10. Given that the training sample size is significantly larger in the MB scheme than under the MR scheme (32,850 pixels compared to 6000 pixels), this suggests that **FRK** v.2 is well suited to fitting and predicting with large sample sizes. Overall, these results suggest that **FRK** v.2 is comparable to other packages in this application. The advantages of **FRK** v.2 lie in the ease with which it allows one to do other more elaborate analyses with non-Gaussian data, as shown in the next sections.

4.2. Block prediction: Contaminated soil

Between 1954 and 1963, nuclear devices were detonated at Area 13 of the Nevada Test Site in the United States, contaminating the surrounding soil with the radioactive element americium (Am). The data we use in this example contains Am concentrations (in 10^3 counts per minute) in a spatial domain immediately surrounding Ground Zero (GZ), the location where the devices were detonated, and was previously analysed by Huang, Yao, Cressie, and Hsing (2009) and Paul and Cressie (2011). The total number of measurements (including some that are collocated) is 212. The left and centre panels of Figure 10 shows the data on the original scale and on the log scale, respectively. Paul and Cressie (2011) note that the Am concentrations are clearly lognormally distributed, and that soil remediation is often made by averaging the contaminant over pre-specified spatial regions of D called blocks. Hence, this

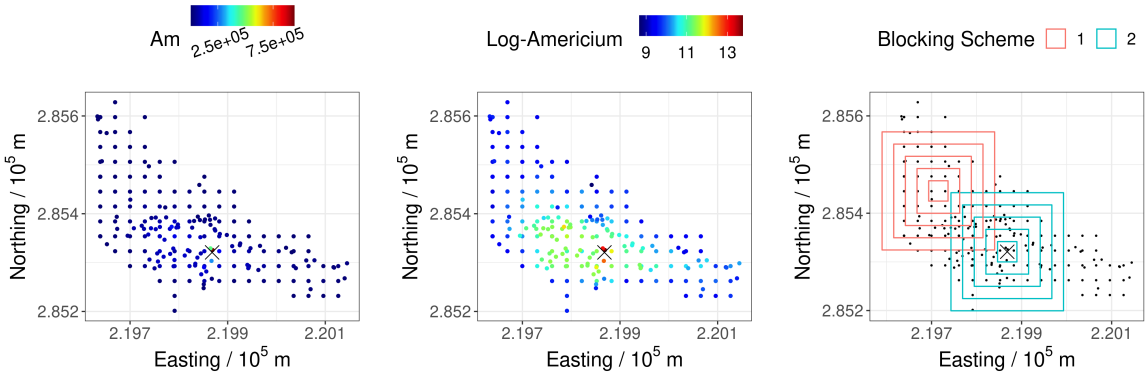

Figure 10: Americium soil data. The '×' denotes Ground Zero (GZ), where the devices were detonated. (Left panel) Am concentrations on the original scale. (Centre panel) Am concentrations on the log scale. (Right panel) Americium soil data blocking schemes: Scheme 1 (red), centred away from GZ, and Scheme 2 (blue), centred on GZ.

application requires lognormal prediction over blocks, a task well suited to **FRK** v.2. The right panel of Figure 10 shows two blocking schemes which we will predict over: Both schemes contain 5 blocks, but one scheme is centred on GZ, and the other is centred away from GZ.

As in Paul and Cressie (2011), we use a piecewise linear trend, where observations within a distance of 30.48m from GZ follow a different trend to those observations beyond 30.48m from GZ. The following code constructs the required BAU level covariates.

```
R> d_BAU   <- distR(coordinates(BAUs), Ground_Zero)
R> BAUs$x1 <- d_BAU * (d_BAU < 30.48)
R> BAUs$x2 <- d_BAU >= 30.48
R> BAUs$x3 <- d_BAU * (d_BAU >= 30.48)
```

Modelling for this problem is done by setting `response = "Gaussian"` and `link = "log"`. In order to mimic lognormal block kriging, here we fix the measurement error standard deviation to a small value prior to model fitting.

```
R> Am_data$std <- 1
R> S <- FRK(f = Am ~ x1 + x2 + x3, data = list(Am_data), BAUs = BAUs,
+    response = "gaussian", link = "log", est_error = FALSE)
```

By predicting over the BAUs, one may generate predictions over the entire spatial domain.

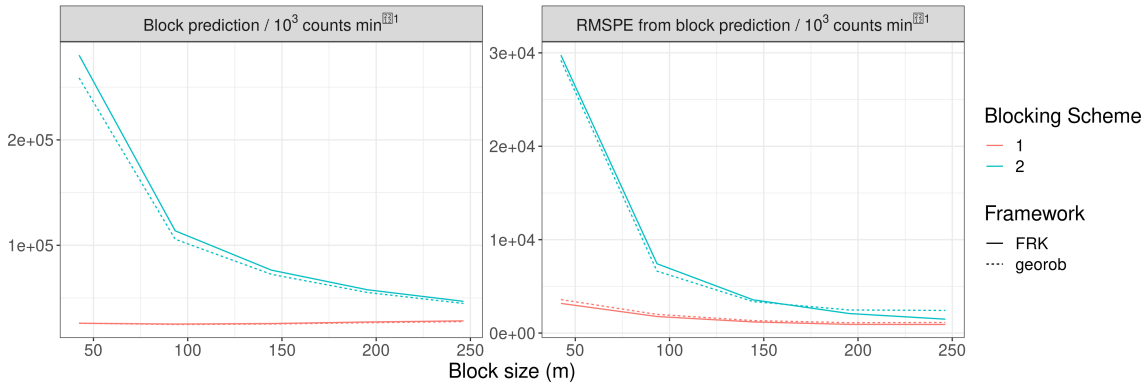

Figure 11: Predictions and root mean squared prediction error (RMSPE) against block size for the two blocking schemes. (Left panel) Block-predictions of Am concentrations against block size $|B|^{1/2}$. (Right panel) RMSPE of block predictions of Am concentrations against block size $|B|^{1/2}$. In both plots, the red line corresponds to Scheme 1 and the blue line corresponds to Scheme 2.

Alternatively, by passing a '`SpatialPolygonsDataFrame`' object into the `newdata` argument of `predict()`, one may straightforwardly generate block-level predictions.

To compare our predictions, we used the R package **georob** (Papritz 2020), which implements an approximately unbiased back-transformation of kriging predictions of log-transformed data (Cressie 2006). Kriging does not scale well for large sample sizes, however the size of this data set is small. The package **georob** provides users with two approaches to lognormal block kriging; we used the 'optimal predictor', as recommended by the **georob** manual when predicting over large blocks. Figure 11 shows the block predictions and associated RMSPE obtained using **FRK** v.2 and **georob** for the two blocking schemes shown in Figure 10. The similarity in results lends confidence that the predictions and associated prediction standard errors obtained using **FRK** v.2 are reasonable.

4.3. Spatial change of support: Poverty in Sydney

The Australian Statistical Geography Standard (ASGS) defines a series of nested geographical areas in Australia known as Statistical Area Levels. Statistical Area Level 3 (SA3) regions are aggregations of Statistical Area Level 2 (SA2) regions, and SA2 regions are aggregations of Statistical Area Level 1 (SA1) regions. In this example, we consider a region of New South Wales containing 7909 SA1 regions, 180 SA2 regions, and 31 SA3 regions, and aim to infer

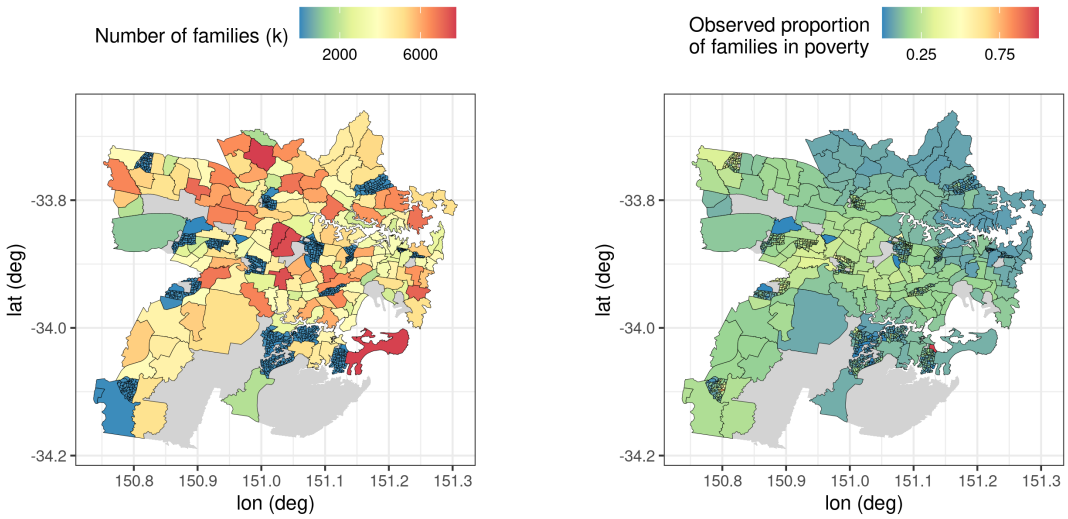

Figure 12: Training data used for modelling the number (or proportion) of families 'in poverty' (see main text for how we define 'in poverty'). (Left panel) The total number of families. (Right panel) The observed proportion of families in poverty, computed by dividing the number of families in poverty by the total number of families. Grey regions correspond to SA regions in which the total number of families is zero.

'poverty' levels at the SA1 and SA3 regions, just from a data set containing mostly SA2 data and a small amount of SA1 data. The data was collected in the Census of 2011, and it consists of the number of families of various types within a range of weekly income brackets; we provide further details, and the way in which we define the poverty line for each family type, in Appendix D. Note that data at the SA1 and the SA3 regions are available, and we use these to validate our down-scaled and up-scaled predictions.

It is often the case that sampling once from a large area is relatively inexpensive compared to acquiring multiple samples from small areas. Our training data, shown in Figure 12, is reflective of such a scenario. It includes mostly SA2 regions, but some SA1 regions have also been included.

In this example, we use the SA2 (and some SA1) region data for training the model, and the SA1 regions as the BAUs; these are passed as '`SpatialPolygonsDataFrame`' objects to `FRK()`. We also set `normalise_wts = FALSE`, which indicates that we wish to model the mean process in a given data polygon as the sum (rather than the average) of the mean process over the SA1s.

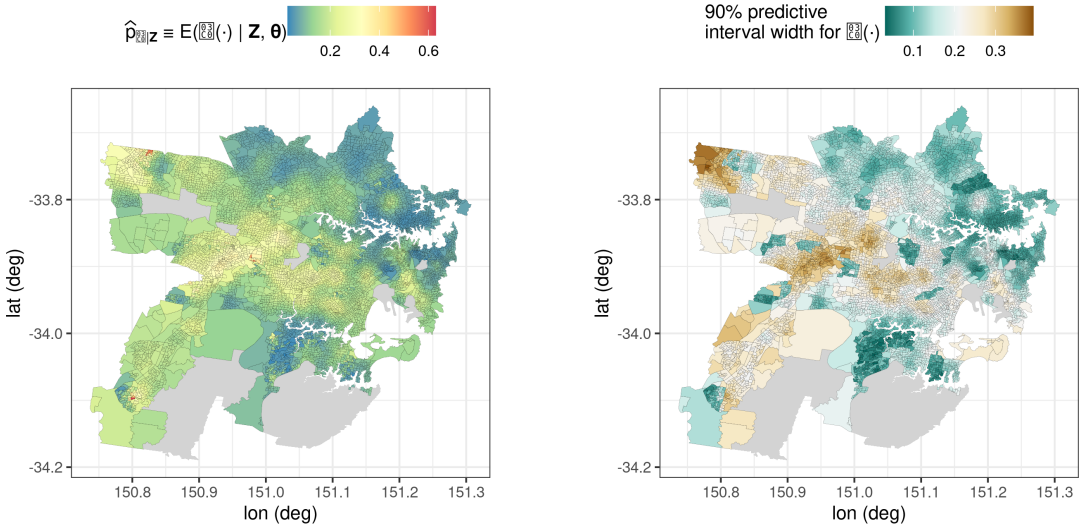

Figure 13: SA1 level predictions. (Left panel) Prediction of the probability process, $\pi(\cdot)$, representing the proportion of families in poverty, over the SA1 regions. (Right panel) 90% prediction interval width of the probability process. Grey regions correspond to SA2 regions in which the total number of families is zero, and hence are omitted from the study.

```
R> S <- FRK(f = total_poverty_count ~ 1,
+     data = list(SA2_and_some_SA1s), BAUs = SA1s,
+     response = "binomial", link = "logit", normalise_wts = FALSE)
```

Now we predict over the SA1 regions.

```
R> SA1_predictions <- predict(S)
```

Since the SA1 regions are the BAUs, we can predict both the probability and mean processes over the SA1 regions: We focus on the probability process, which is independent of the size parameter. The predictions and associated uncertainty over the SA1 regions are shown in Figure 13, which was generated using `plot(S, SA1_predictions)`.

Predicting over different spatial supports is straightforward with **FRK** v.2. To predict over the SA3 regions, we simply set `newdata` to a '`SpatialPolygonsDataFrame`' object containing the SA3 regions.

```
R> SA3_predictions <- predict(S, newdata = SA3s)
```

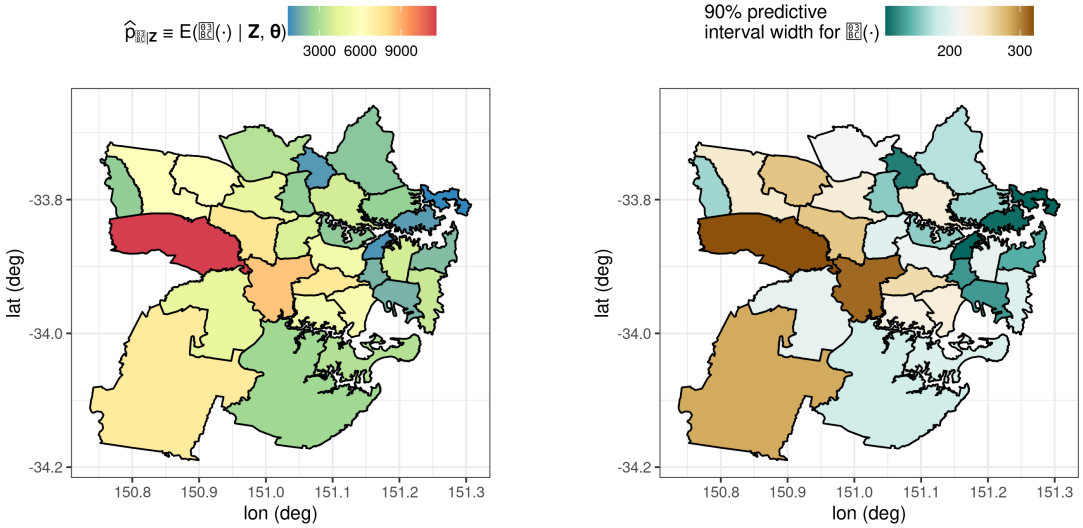

Figure 14: SA3 level predictions. (Left panel) Prediction of the mean process, $\mu(\cdot)$, representing the expected number of families in poverty, over the SA3 regions. (Right panel) The 90% prediction interval width of the mean process.

Figure 14 shows the SA3 region predictions and associated uncertainty quantification: Since the SA3 regions are not the BAUs, this time we are restricted to prediction of the mean process. Again, this graphic was generated with `plot(S, SA3_predictions)`.

We assessed the models ability quantify uncertainty over the SA1 regions by computing the empirical coverage from 90% prediction intervals. The empirical coverage was 90.8%, which is almost nominal. The inclusion of some fine-scale data (SA1 region data) greatly aids in the estimation of the fine-scale variance parameter, σ^2_ξ. If only coarse-resolution data is available (i.e., all data supports are associated with multiple BAUs), in order to avoid identifiability issues, **FRK** v.2 fixes σ^2_ξ prior to model fitting with **TMB**. In this situation, if σ^2_ξ is unknown, **FRK** v.2 generates a rough and possibly unreliable estimate. If one does know σ^2_ξ, or can obtain a reliable estimate of it (for example, using past census data), one may specify it using the argument `known_sigma2fs`.

We conclude this example by noting that, although the prediction polygons, data supports, and BAUs in this study have a nested relationship, in general these elements can be entirely unrelated: Prediction in **FRK** can be done over any arbitrary, user-specified polygons. See Section 3.2 for an illustration.

4.4. Non-Gaussian spatio-temporal data: Crime in Chicago

The city of Chicago is divided into 77 so-called community areas (CAs). An attractive property of CAs is their relative consistency, their boundaries having changed little since their inception in the 1920's (The University of Chicago Library 2020). In this study, we model the number of crimes in each CA between the years 2001 and 2019. A full list of crimes committed in Chicago during this period is provided by the Chicago Police Department, and is available for download from the open data source website Plenario (Urban Center for Computation and Data and University of Chicago 2020). We considered crimes labelled as assault or battery; roughly 1.75 million crimes in total. Note that **FRK** bins data falling into the same BAU, so the final number of observations post-binning is significantly less. The CA containing O'Hare airport is non-populous and is almost disjoint from the other CAs; for simplicity, we excluded it from this analysis.

In this example, we use the CAs as our spatial BAUs. This can be done straightforwardly by reading in the shapefile of the CAs as a '`SpatialPolygonsDataFrame`' object. Spatio-temporal BAUs may then be constructed by passing the CAs and data into `auto_BAUs()`.

```
R> ST_BAUs <- auto_BAUs(manifold = STplane(), data = chicago_crimes_fit,
+    spatial_BAUs = community_areas, tunit = "years")
```

When modelling crime, it is natural to include population, or population density, as a covariate. As the CAs are of unequal area, we use population rather than population density. This covariate was obtained from the Combined Community Data Snapshots provided by the Chicago Metropolitan Agency for Planning (2017). It is difficult to obtain population data for every year, so we assume that population is constant over the time-span of the data. We observed a distinct negative trend when plotting the total number of crimes in each year; hence, we also include time as a covariate. The required BAU level covariates may be constructed in an analogous fashion to the way in which the covariates were constructed in Section 4.2. Next, we generate spatio-temporal basis functions automatically using `auto_basis()`.

```
R> basis <- auto_basis(STplane(), chicago_crimes_fit, tunit = "years")
```

Then, we initialise and fit the 'SRE' object using `FRK()`, setting `response = "poisson"` and `link = "log"`. Each entry of our data provides the location and time at which a given crime occurred. It also contains a column of ones called `"number_of_crimes"`; this will be used for binning. By default, `SRE()`, which is called internally within `FRK()`, bins and then averages data falling into the same BAU. We wish to model the total number of crimes in a given BAU; hence, we wish to sum the binned data instead of average. To do so, we pass the name of the response variable (`"number_of_crimes"`) via the argument `sum_variables`. As the number of spatial BAUs (the CAs) is relatively low, and we have observed each spatial BAU multiple times, we may attribute each spatial BAU its own fine-scale variance parameter (see Section 2.5).

```
R> S <- FRK(f = number_of_crimes ~ log(population) + x1 + x2 + x3,
+    data = list(chicago_crimes_fit), basis = basis, BAUs = ST_BAUs,
+    response = "poisson", link = "log",
+    sum_variables = "number_of_crimes", fs_by_spatial_BAU = TRUE)
```

Finally, we predict over the spatio-temporal BAUs (which use the CAs as spatial BAUs) using `predict()`, and plot the results using `plot()`.

To validate predictions, we excluded the years 2010 and 2019 from the training data. The observed number of crimes, predicted number of crimes, and prediction uncertainty for these years are provided in Figure 15. For both years, Figure 15 shows agreement between the predicted and observed number of crimes. Furthermore, the prediction uncertainty is roughly proportional to the predicted value, as one may expect when modelling counts. For these validation years, we also computed the empirical coverage when using 90%, 80%, 70%, and 60% posterior predictive intervals, and the mean absolute percentage error (MAPE; see Appendix C). We consistently observed that the empirical coverage in the year 2010 was slightly (4% on average) higher than the purported coverage, whilst it was lower (16% on average) in the forecast year. We observed MAPE scores of 4.4% and 9.0% in the years 2010 and 2019, respectively. The slightly worse results in the year 2019 is possibly expected, as forecasting into the future is, in general, a harder task than predicting within the time-span of the data.

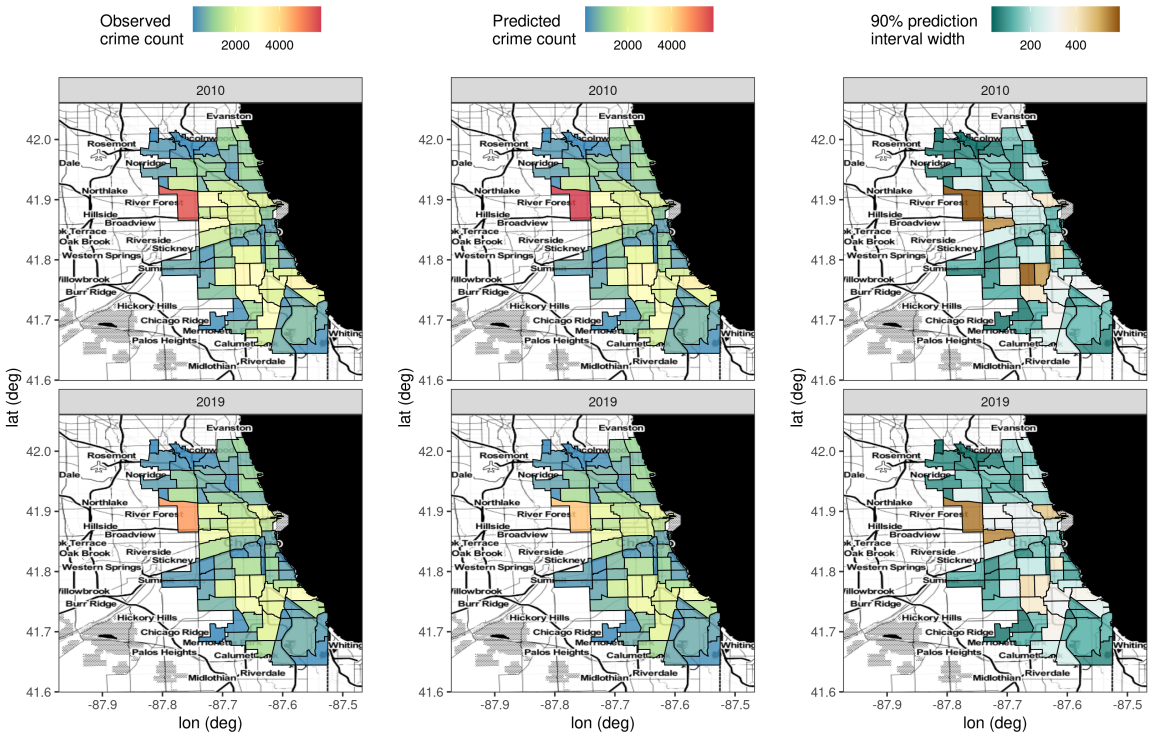

Figure 15: Observed number of crimes, predictions, and prediction uncertainty over Chicago in the prediction (2010) and forecast (2019) years. The first row corresponds to the year 2010; the second row corresponds to the year 2019. The first column shows the observed number of crimes; the second column shows the predicted number of crimes; the third column shows the width of a posterior predictive interval with a purported coverage of 90%.

Nonetheless, these predictions are cause for optimism given the difficulties in modelling crime in a spatio-temporal setting.

For illustration, we chose three CAs of interest: Ashburn, Roseland, and Archer Heights. The time-series of the observed data, predictions, and 90% posterior predictive intervals for these CAs is shown in Figure 16. The posterior predictive intervals are slightly wider in validation years (2010 and 2019) than in observed years. The observed number of crimes is contained within the predictive interval for all time-points for these CAs. The predictive distributions in the validation years for the three CAs of interest is shown in Figure 17. The forecasts for Ashburn and Roseland in the year 2019 are particularly accurate, with the predicted number of crimes essentially equal to the observed number of crimes.

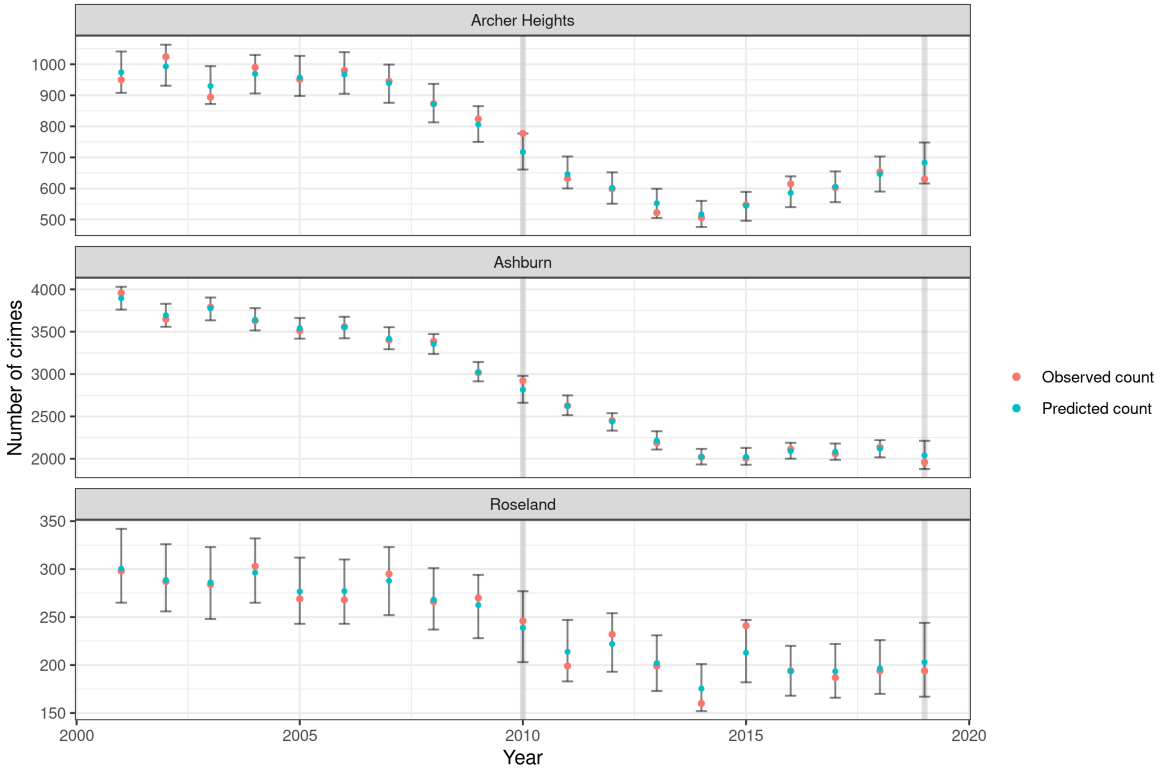

Figure 16: Time-series plots of predictions and observed number of crimes for three CAs of interest. The validation years, 2010 and 2019, are highlighted in light-grey. The observed number of crimes at each time is indicated by a red point, whilst the predicted number of crimes is indicated by a blue point. The error bars represent a 90% posterior predictive interval. We note that the posterior predictive intervals are slightly wider in validation years (2010 and 2019) than in observed years, and that the observed crime is contained within the predictive interval for all time-points for these CAs.

5. Conclusion

In this paper we have described an extension to the R package **FRK** which allows for the spatial and spatio-temporal modelling and prediction of big, non-Gaussian data. Thanks to the use of a GLMM model and the software **TMB**, **FRK** v.2 can now cater for many distributions within the exponential family, and many link functions. Furthermore, **FRK** v.2 allows for the use of many more basis functions when modelling the spatial process, and can therefore also often achieve more accurate predictions in a Gaussian setting than **FRK** v.1. The existing functionality of **FRK** is retained with this extension; in particular, the package makes use of automatic basis function construction, is capable of handling both point-referenced and

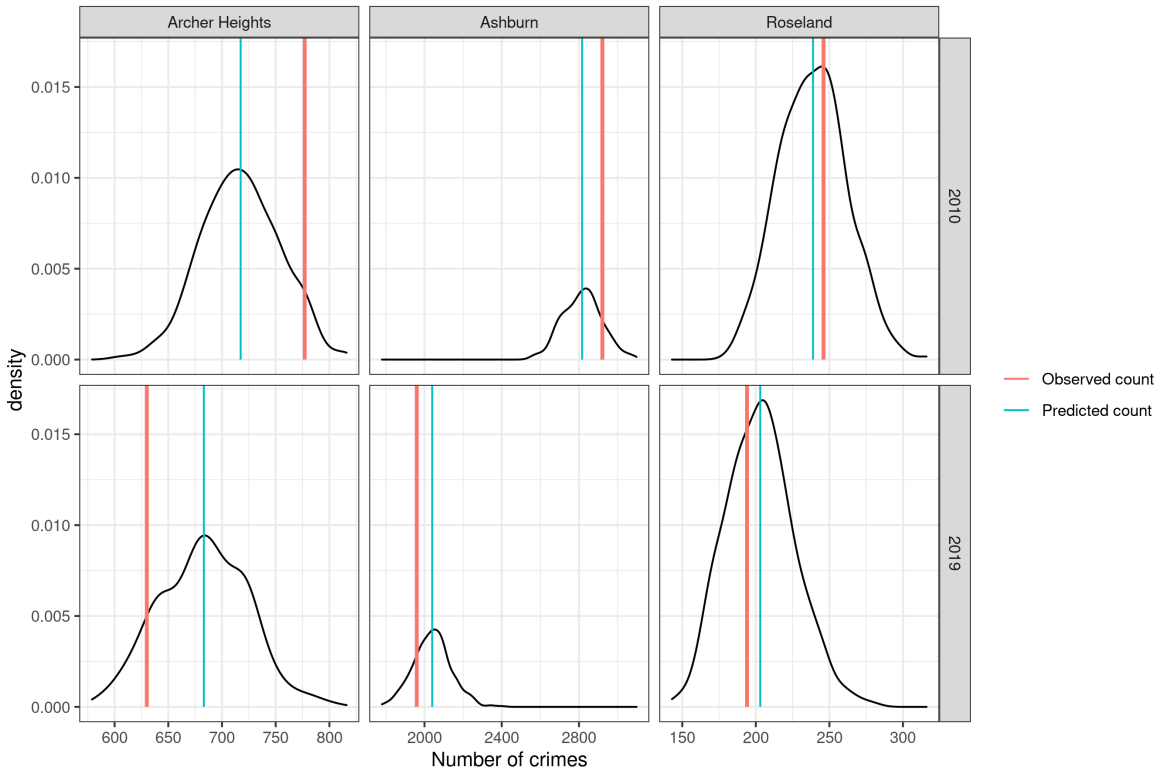

Figure 17: Posterior predictive distributions in the validation years (2010 and 2019) for three CAs (Archer Heights, Ashburn, and Roseland). The red and blue lines correspond to the observed and predicted number of crimes at each CA in the given year. The first *row* corresponds to the year 2010; the second row corresponds to the year 2019. The first *column* corresponds to Archer Heights; the second column corresponds to Ashburn; the third column corresponds to Roseland.

areal data, and eases the so-called spatial change-of-support problem through the use of BAUs. The package now provides a highly accessible and user friendly approach to spatial and spatio-temporal modelling of big data in both a Gaussian and non-Gaussian setting.

One limitation of the framework is that it requires covariates to be known for every BAU, which may not be the case if covariates are recorded only at the data support level. Another limitation is the necessity to fix the fine-scale variance parameter in spatial change-of-support applications when `method = "TMB"`; note that this is not an issue if one is able to obtain a reliable estimate through other means (e.g., via previous census data). We are currently exploring avenues to address this limitation, either through adjusting the model to allow estimation of the fine-scale variance parameter within **TMB**, or via a more robust offline

estimate. Another limitation is that despite the added flexibility, several models of interest, such as the zero-inflated Poisson, are still not catered for. The introduction of different types of models is facilitated by **TMB**'s implementation of automatic differentiation, which means we can make use of existing code within the C++ template straightforwardly; future work will see the introduction of other models of interest. The main spatial data structures used in **FRK** come from the package **sp** (Pebesma and Bivand 2005); future work may entail the inclusion of other spatial data structures, such as those from the package **sf** (Pebesma 2018).

Acknowledgments

Matthew Sainsbury-Dale's research was supported by an Australian Government Research Training Program Scholarship. Andrew Zammit-Mangion's and Noel Cressie's research was supported by an Australian Research Council (ARC) Discovery Project, DP190100180. Andrew Zammit-Mangion's research was also supported by an ARC Discovery Early Career Research Award, DE180100203. The authors would like to thank Rajib Paul for providing the Americium data analysed in Section 4.2, and Michael Bertolacci for discussion surrounding the MODIS comparison study.

References

Bachl FE, Lindgren F, Borchers DL, Illian JB (2019). "inlabru: an R package for Bayesian spatial modelling from ecological survey data." *Methods in Ecology and Evolution*, **10**, 760–766. doi:10.1111/2041-210X.13168.

Bates D, Maechler M, Davis TA (2019). *Matrix: Sparse and Dense Matrix Classes and Methods*. R package version 1.2-17, URL http://Matrix.R-forge.R-project.org/.

Bell BM (2005). "CppAD: a package for C++ algorithmic differentiation." http://www.coin-or.org/CppAD. Accessed: 2019-06-15.

Bivand R (2020). "CRAN Task View: Analysis of Spatial Data." https://CRAN.R-project.org/view=Spatial. Accessed: 2020-04-03.

Chicago Metropolitan Agency for Planning (2017). "Chicago community data snapshots." https://www.cmap.illinois.gov/documents/10180/126764/_Combined_AllCCAs.pdf/. Accessed: 2020-09-18.

Cressie N (1993). *Statistics for Spatial Data*. revised edition. John Wiley & Sons, Hoboken, NJ.

Cressie N (2006). "Block kriging for lognormal spatial processes." *Mathematical Geology*, **38**, 413–443.

Cressie N, Johannesson G (2008). "Fixed rank kriging for very large spatial data sets." *Journal of the Royal Statistical Society*, **70**, 209–226.

Datta A, Banerjee S, Finley AO, Gelfand AE (2016). "Hierarchical nearest-neighbour Gaussian process models for large geostatistical datasets." *Journal of the American Statistical Association*, **111**, 800–812.

Diggle PJ, Tawn JA, Moyeed RA (1998). "Model-based geostatistics." *Journal of the Royal Statistical Society*, **47**, 299–350.

Finley AO, Banerjee S, Gelfand AE (2015). "`spBayes` for large univariate and multivariate point-referenced spatio-temporal data models." *Journal of Statistical Software*, **63**, 1–28. URL http://www.jstatsoft.org/v63/i13/.

Finley AO, Datta A, Banerjee S (2020). "`spNNGP R` package for nearest neighbour Gaussian process models." *arXiv:2001.09111*.

Furrer R, Nychka D, Genton MG (2006). "Covariance tapering for interpolation of large spatial datasets." *Journal of Computational and Graphical Statistics*, **15**, 502–523. doi:10.1198/106186006X132178.

Gneiting T, Balabdaoui F, Raftery AE (2007). "Probabilistic forecasts, calibration and sharpness." *Journal of the Royal Statistical Society*, **69**, 243–268.

Guennebaud G, Jacob B, *et al.* (2010). "`Eigen` v3." http://eigen.tuxfamily.org. Accessed: 11-05-2019.

Heaton MJ, Datta A, Finley AO, Furrer R, Guinness J, Guhaniyogi R, Gerber F, Gramacy RB, Hammerling D, Katzfuss M, Lindgren F, Nychka DW, Sun F, Zammit-Mangion A (2019). "A case study competition among methods for analyzing large spatial data." *Journal of Agricultural, Biological and Environmental Statistics*, **24**, 398–425.

Hersbach H (2000). "Decomposition of the continuous ranked probability score for ensemble prediction systems." *American Meteorological Society*, **15**, 559–570.

Huang C, Yao Y, Cressie N, Hsing T (2009). "Multivariate intrinsic random functions for cokriging." *International Association for Mathematical Geosciences*, **41**, 887–904.

Kassambara A (2020). *ggpubr: 'ggplot2' Based Publication Ready Plots*. R package version 0.4.0, URL https://CRAN.R-project.org/package=ggpubr.

Kristensen K, Nielsen A, Berg CW, Skaug H, Bell BM (2016). "TMB: Automatic differentiation and Laplace approximation." *Journal of Statistical Software*, **70**, 1–21.

Lee BS, Park J (2020). "A scalable partitioned approach to model massive nonstationary non-Gaussian spatial datasets." *arXiv:2001.09111*.

Lindgren F, Rue H (2011). "An explicit link between Gaussian fields and Gaussian Markov random fields: The stochastic partial differential equation approach." *Journal of the Royal Statistical Society*, **73**, 423–498.

Lindgren F, Rue H (2015). "Bayesian spatial modelling with R-INLA." *Journal of Statistical Software*, **63**, 1–25.

Lopes HF, Gamerman D, Salazar E (2011). "Generalized spatial dynamic factor models." *Computational Statistics and Data Analysis*, **55**, 1319–1330.

McCullagh P, Nelder JA (1989). *Generalized Linear Models*. Chapman & Hall, London, UK.

MODIS Characterization Support Team (2015). "MODIS 500m Calibrated Radiance Product. NASA MODIS Adaptive Processing System, Goddard Space Flight Center, USA." https://mcst.gsfc.nasa.gov/.

Nychka D, Hammerling D, Sain S, Lenssen N (2016). *LatticeKrig: Multiresolution Kriging Based on Markov Random Fields*. R package version 6.2, URL www.image.ucar.edu/LatticeKrig.

Papritz A (2020). *georob: Robust Geostatistical Analysis of Spatial Data*. R package version 0.3-13, URL https://cran.r-project.org/web/packages/georob/index.html.

Paul R, Cressie N (2011). "Lognormal block kriging for contaminated soil." *European Journal of Soil Science*, **62**, 337–345.

Pebesma E (2018). "Simple Features for R: Standardized Support for Spatial Vector Data." *The R Journal*, **10**, 439–446. doi:10.32614/RJ-2018-009. URL https://doi.org/10.32614/RJ-2018-009.

Pebesma E (2020). "CRAN Task View: Handling and Analyzing Spatio-Temporal Data." https://CRAN.R-project.org/view=SpatioTemporal. Accessed: 2020-04-03.

Pebesma EJ, Bivand RS (2005). "Classes and methods for spatial data in R." *R News*, **5**, 9–13. URL https://CRAN.R-project.org/doc/Rnews/.

R Core Team (2020). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

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*, **71**, 3319–392.

Sengupta A, Cressie N (2013). "Hierarchical statistical modelling of big spatial datasets using the exponential family of distributions." *Spatial Statistics*, **4**, 14–44.

Sengupta A, Cressie N (2016). "Predictive inference for big, spatial, non-Gaussian data: MODIS cloud data and its change-of-support." *Australian & New Zealand Journal of Statistics*, **58**, 15–45.

The University of Chicago Library (2020). "Spatially Referenced Census Data for the City of Chicago: Sources Available at or through the University of Chicago Library." https:

//www.lib.uchicago.edu/e/collections/maps/censusinfo.html. Accessed: 2020-09-15.

Urban Center for Computation and Data and University of Chicago (2020). "Plenario." http://plenar.io/explore/discover. Accessed: 11-09-2020.

Wickham H (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. ISBN 978-3-319-24277-4. URL https://ggplot2.tidyverse.org.

Wood S (2017). *Generalized Additive Models: An Introduction with R*. 2nd edition. Chapman and Hall/CRC, Boca Raton, FL.

Zammit-Mangion A, Cressie N (2021). "FRK: an R package for spatial and spatio-temporal prediction with large datasets." *Journal of Statistical Software*, **In press**.

Zammit-Mangion A, Ng TLJ, Vu Q, Filippone M (2021). "Deep compositional spatial models." *Journal of the American Statistical Association*, **In press**.

A. Parametrisations of the basis-function coefficients

Recall from Section 2.1 that **FRK** v.2 allows the basis-function coefficients, $\boldsymbol{\eta}$, to be parametrised using either a prior covariance matrix, $\boldsymbol{K}$, or using a prior precision matrix, $\boldsymbol{Q}$. In this appendix, we describe these matrices. Recall that both formulations use block-diagonal matrices, wherein basis-function coefficients are independent of the basis-function coefficients in differing resolutions. Hence, we need only describe the intra-resolution dependencies.

A.1. Covariance matrix

Let $K_k(\boldsymbol{s}, \boldsymbol{s}^*)$ denote the covariance function of the basis-function coefficients corresponding to the kth basis function resolution. In **FRK**, we model $K_k(\boldsymbol{s}, \boldsymbol{s}^*)$ using the exponential covariance function,

$$K_k(\boldsymbol{s}, \boldsymbol{s}^*) = \sigma_k^2 \exp\left\{\frac{-d(\boldsymbol{s}, \boldsymbol{s}^*)}{\tau_k}\right\}, \tag{A.1}$$

where $d(\boldsymbol{s}, \boldsymbol{s}^*)$ is the distance between locations $\boldsymbol{s}, \boldsymbol{s}^* \in D$. Clearly (A.1) is always non-zero for $\sigma_k^2 > 0$, however it is often reasonable to assume that coefficients associated with fine-resolution basis functions separated by medium-to-large distances are uncorrelated. To increase sparsity, **FRK** v.2 allows covariance tapering (Furrer, Nychka, and Genton 2006) of the intra-resolution covariance function. Noting that (A.1) is a special case of the Matérn covariance function with $\nu = 0.5$, we follow the recommendation of Furrer *et al.* (2006) and use the Spherical taper:

$$T_{\beta_k}(\boldsymbol{s}, \boldsymbol{s}^*) = \left\{1 - \frac{d(\boldsymbol{s}, \boldsymbol{s}^*)}{\beta_k}\right\}_+^2 \left\{1 + \frac{d(\boldsymbol{s}, \boldsymbol{s}^*)}{2\beta_k}\right\}, \tag{A.2}$$

where $x_+ \equiv \max(0, x)$, and β_k is a resolution-dependent tapering parameter controlling the strength of the taper. In **FRK** v.2, we define β_k based on the minimum distance between basis-function centroids; specifically, we set $\beta_k = \texttt{taper} \times \text{mindist}(k)$, where mindist$(k)$ is the minimum distance between the centroids of basis functions at the kth resolution and `taper` is a user-specified argument. The tapered covariance function is obtained by taking the product

of the original covariance function (A.1) and the taper function (A.2).

A.2. Precision matrix

FRK v.2 offers two types of sparse precision matrices: One for regularly spaced basis functions, and another for irregularly spaced basis functions. This choice is determined by the field `regular` in the '`Basis`' object.

When the basis functions are regularly spaced (`regular = TRUE`), **FRK** v.2 uses a precision matrix that is related to that used in the R package **LatticeKrig** (Nychka, Hammerling, Sain, and Lenssen 2016). Let $\mathcal{N}_{i,k}$ denote the set of first-order horizontal and vertical neighbouring basis functions of the ith basis function of resolution k, and let $\boldsymbol{Q}_k$ denote the prior precision matrix of the basis-function coefficients at resolution k.

We model the elements of $\boldsymbol{Q}_k$ as

$$\{\boldsymbol{Q}_k\}_{i,j} = \begin{cases} \kappa_k + \rho_k |\mathcal{N}_{i,k}| & i = j \\ -\rho_k & j \in \mathcal{N}_{i,k} \\ 0 & \text{otherwise} \end{cases}, \tag{A.3}$$

where κ_k and ρ_k are parameters that need to be estimated. We note that $\boldsymbol{Q}_k$ is diagonally dominant, and hence it is positive definite. This formulation implies that the coefficient of a given basis-function is conditionally independent of all other basis-function coefficients given the coefficients of its first-order vertical and horizontal neighbours. We also note that **LatticeKrig** uses $\boldsymbol{Q}_k^\top \boldsymbol{Q}_k$ as the precision matrix blocks.

To cater for irregularly spaced basis functions, **FRK** v.2 also offers a sparse precision matrix which considers the distance between basis functions:

$$\{\boldsymbol{Q}_k\}_{i,j} = \begin{cases} \kappa_k - \sum_{j \neq i} \{\boldsymbol{Q}_k\}_{i,j} & i = j \\ -\rho_k \exp\left\{ \frac{-d(\boldsymbol{s}_{i,k}, \boldsymbol{s}_{j,k})}{\tau_k} \right\} T_{\beta_k}(\boldsymbol{s}_{i,k}, \boldsymbol{s}_{j,k}) & i \neq j \end{cases}, \tag{A.4}$$

where κ_k, ρ_k, and κ_k are parameters that need to be estimated, and $T_{\beta_k}(\cdot, \cdot)$ is defined as in

the preceding section. Again, this matrix is diagonally dominant, and hence positive definite. Equation (A.4) is in some ways a generalisation of (A.3). This formulation implies that the partial correlation between basis-function coefficients decays exponentially with distance until a point (controlled by the tapering parameter β_k) at which the basis-function coefficients are conditionally independent.

B. Distributions with size parameters

Two data models that can be used with **FRK** v.2, namely, the binomial and negative-binomial distributions, have a known constant 'size' parameter k_j and a 'probability of success' parameter, π_j, associated with every datum Z_j. For binomial data models, k_j represents the number of trials, and Z_j the number of successes; for negative-binomial data models, k_j represents the target number of successes, and Z_j the number of failures.

Consider a negative-binomial data model with a logit-link function; under the standard interpretation of a link function (a function which transforms the mean $\mu(\cdot)$ to the linear predictor $Y(\cdot)$), one models

$$g(\mu(\cdot)) = \text{logit}(\mu(\cdot)) = Y(\cdot).$$

In this example, the range of the mean function (the inverse link function, $g^{-1}(\cdot)$) is $(0, 1)$. However, the mean of negative-binomial distribution may take values in $[0, \infty)$. Direct use of the logit link would thus unacceptably restrict the range of the mean function.

Therefore, for both the binomial and negative-binomial distributions, we first model $\pi(\cdot)$ as a function of $Y(\cdot)$, and then link $\pi(\cdot)$ to the mean $\mu(\cdot)$. That is, we use a hierarchical link function;

$$f(\pi(\cdot)) = Y(\cdot),$$
$$h(\mu(\cdot); k) = \pi(\cdot),$$

where $h(\cdot)$ is a function determined solely by the response distribution, and $f(\cdot)$ is a function

that maps $\pi(\cdot)$ to the latent process $Y(\cdot)$. The implied link function is

$$g(\mu(\cdot); k) = f(h(\mu(\cdot); k)) = (f \circ h)(\mu(\cdot); k) = Y(\cdot).$$

Using a hierarchical link function approach with a negative-binomial data model and a logit-link function, we have that

$$f(\pi(\cdot)) = \text{logit}(\pi(\cdot)) = Y(\cdot),$$

and, as the expectation of the negative-binomial distribution in terms of the probability of success is $\mu(\cdot) = k\left(\frac{1}{\pi(\cdot)} - 1\right)$, we have that

$$h(\mu(\cdot)) = \frac{k}{\mu(\cdot) + k} = \pi(\cdot).$$

Observe that $\pi(\cdot) \in (0, 1)$, so that $\mu(\cdot) \in (0, \infty)$. Hence, in **FRK** v.2, whenever the data model is specified to be binomial or negative-binomial, and a logit, probit, or complementary log-log 'link' is specified, we use it to define $f(\cdot)$ and to transform the probability parameter. We then map the probability parameter to the mean of the data using the known form of the mean specific to the distribution in question via $h(\cdot)$. If the user specifies a link function that is not appropriate for modelling probability parameters (such as the log or square-root link), then we use this to define $g(\cdot)$ directly, whilst also accounting for the size parameter; specifically, we set $g(\mu(\cdot)/k) = Y(\cdot)$.

C. Scoring rules

Suppose that we have a validation domain $D^* \subset D$ which is used for model validation. As prediction-performance measures for the examples in this paper, we considered the following. For simplicity, we describe the measures in terms of prediction of the mean process.

- (Empirical) root-mean-squared prediction error (RMSPE): Let $\hat{\mu}(\boldsymbol{s})$ denote a point-predictor of $\mu(\boldsymbol{s})$, where $\mu(\boldsymbol{s})$ is the true value of the mean process evaluated at location

$\boldsymbol{s}$. Then the (empirical) RMSPE, used to assess point-wise predictive performance, is

$$\text{RMSPE} \equiv \sqrt{\frac{1}{|D^*|} \sum_{\boldsymbol{s} \in D^*} (\hat{\mu}(\boldsymbol{s}) - \mu(\boldsymbol{s}))^2}.$$

- (Empirical) mean-absolute error (MAE): The mean-absolute error, also used to assess point-wise predictive performance, is

$$\text{MAE} \equiv \frac{1}{|D^*|} \sum_{\boldsymbol{s} \in D^*} |\hat{\mu}(\boldsymbol{s}) - \mu(\boldsymbol{s})|.$$

- (Empirical) mean-absolute percentage error (MAPE): The mean-absolute percentage error, is similar to the MAE, but we also divide by the true value;

$$\text{MAPE} \equiv \frac{1}{|D^*|} \sum_{\boldsymbol{s} \in D^*} \left| \frac{\hat{\mu}(\boldsymbol{s}) - \mu(\boldsymbol{s})}{\mu(\boldsymbol{s})} \right|.$$

- Continuous ranked probability score (CRPS; Gneiting *et al.* 2007, sec 4.2.): Let $F(\mu; \boldsymbol{s}, \boldsymbol{Z})$ denote the posterior predictive cumulative distribution function (CDF) of the mean process at location $\boldsymbol{s}$. The CRPS is used to evaluate a predictive CDF, and is defined as

$$\text{CRPS}(F, \mu(\boldsymbol{s})) \equiv \frac{1}{|D^*|} \sum_{\boldsymbol{s} \in D^*} \int_{-\infty}^{\infty} (F(u; \boldsymbol{s}, \boldsymbol{Z}) - \mathbb{1}\{u \geq \mu(\boldsymbol{s})\})^2 \mathrm{d}u,$$

  where $\mathbb{1}\{\cdot\}$ denotes an indicator function that takes the value 1 if its argument is true, and 0 otherwise. For some predictive CDFs (in particular, the Gaussian and log-normal) there exist closed form expressions to compute the CRPS; however, in general no closed form expression exists, in which case we may use an *empirical* predictive CDF from a sample (e.g., a Monte Carlo sample) to evaluate the CRPS in terms of the respective order statistics (Hersbach 2000).

- Interval score (Gneiting *et al.* 2007, sec. 6.2): The interval score for a purported $(1 -$

$\alpha) \times 100\%$ prediction interval is defined as

$$S_\alpha^{\text{int}} \equiv \frac{1}{|D^*|} \sum_{\boldsymbol{s} \in D^*} \Big(u(\boldsymbol{s}) - l(\boldsymbol{s}) + \\ \frac{2}{\alpha}(l(\boldsymbol{s}) - \mu(\boldsymbol{s}))\mathbb{1}\{\mu(\boldsymbol{s}) < l(\boldsymbol{s})\} + \frac{2}{\alpha}(\mu(\boldsymbol{s}) - u(\boldsymbol{s}))\mathbb{1}\{\mu(\boldsymbol{s}) > u(\boldsymbol{s})\} \Big),$$

where $l(\boldsymbol{s})$ and $u(\boldsymbol{s})$ are the lower and upper bounds of the prediction interval at location $\boldsymbol{s}$. It rewards narrow prediction intervals, and penalises instances in which an observation misses the interval (with the size of the penalty depending on α).

- Coverage: The coverage of a prediction interval is defined as

$$\text{Cvg} \equiv \frac{1}{|D^*|} \sum_{\boldsymbol{s} \in D^*} \mathbb{1}\{l(\boldsymbol{s}) \leq \mu(\boldsymbol{s}) \leq u(\boldsymbol{s})\}$$

  If the interval is indeed a $(1 - \alpha) \times 100\%$ prediction interval, the coverage should be approximately equal to $1 - \alpha$.

- Brier score (Gneiting *et al.* 2007, sec 3.): The Brier score, applicable in a binary setting, is defined as

$$\text{Brier Score} \equiv \frac{1}{|D^*|} \sum_{\boldsymbol{s} \in D^*} (Z_{\boldsymbol{s}} - \hat{\pi}(\boldsymbol{s}))^2,$$

  where $Z_{\boldsymbol{s}}$ denotes the validation data (taking a value of 0 or 1), and $\hat{\pi}(\boldsymbol{s})$ denotes a point-prediction of the probability process, at location $\boldsymbol{s}$.

D. Sydney poverty lines

Here we provide some details on how we defined the poverty line for the data in Section 4.3. Recall that our data consists of the number of families of various types (these types are 'couple family with no children', 'couple family with children','one parent family', and 'other family') within a range of weekly income brackets. For families with a weekly income below \$1000 (which, as we explain subsequently, are the families of interest for this analysis), the income brackets are defined in intervals of \$200: negative or nil income, [\$1–\$199],

[$200–$399], ..., [$800–$999]. To determine the number of families 'in poverty' within each Statistical Area, we must define poverty lines. The Melbourne Institute of Applied Economic and Social Research (MIAESR) provided poverty line guidelines for a range of family structures in March 2011 (https://melbourneinstitute.unimelb.edu.au/assets/documents/poverty-lines/2017/Poverty-Lines-Australia-March-Quarter-2011.pdf). Unfortunately, the groupings of our family units do not align exactly with the poverty line definitions as given by the MIAESR, and so, since this example is shown for purely illustrative purposes, we make several assumptions. First, we assume 'families with children' consist of exactly two parents and two children. Second, since 'other families' is difficult to interpret and categorise appropriately in the context of the MIAESR guidelines, we exclude 'other families' from the study (less than 2% of all families). Third, our data do not make clear whether the head of the family is in the workforce; we therefore assume that the head of the family *is* in the workforce, and hence use the first half of Table 1 of the MIAESR guidelines. Fourth, our data do not provide exact income figures, but rather income brackets of width $200; we thus round MIAESR guidelines to the nearest $200. Hence, the definition of poverty lines (in Australian dollars) for each family unit considered in this study are the weekly incomes of: $600 for a couple with no children, $800 for a couple with children, and $600 for a one parent family. We do not consider SA2 regions where the total number of families (the size parameter) is equal to zero, as this would cause issues with model fitting. (Note that there is nothing in the framework preventing us from *predicting* over these 'empty' SA2 regions, however, for interpretability reasons, we choose not to do so.)

Affiliation:

Matthew Sainsbury-Dale, Andrew Zammit-Mangion, Noel Cressie

National Institute for Applied Statistics Research Australia (NIASRA)

School of Mathematics and Applied Statistics

University of Wollongong

Wollongong, Australia

E-mail: msdale@uow.edu.au

URL: https://github.com/MattSainsbury-Dale

Journal of Statistical Software http://www.jstatsoft.org/

published by the Foundation for Open Access Statistics http://www.foastat.org/

MMMMMM YYYY, Volume VV, Issue II *Submitted:* yyyy-mm-dd

doi:10.18637/jss.v000.i00 *Accepted:* yyyy-mm-dd