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 **?**, is an R (**?**) 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* (**?**, 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 **?**, 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. **?** used the spatial GLMM framework, but with a reduced-rank factor analytic model for $Y(\cdot)$. **?** 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. **?** 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 (**?**). 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. **?** 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' (**?**) and 'Handling and Analyzing Spatio-Temporal Data' (**?**). A popular general purpose R package is **INLA** (**?**), which uses the integrated nested Laplace Approximation (**?**) 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 (**?**), 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** (**?**). 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** (**?**) 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** (**?**) facilitates dimension reduction using a nearest neighbour Gaussian Process (**?**), 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 **??**, we establish the proposed statistical framework, and describe model fitting and prediction using the R package **TMB** (**?**). In Section **??**, 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 **??**, 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 **??** concludes.

2. Methodology

The statistical model used in **FRK** v.2 is a spatial GLMM (**?**), 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 (**??**); see Section **??**. In the *data layer*, we use a conditionally independent exponential-family model for the data; see Section **??**. In Section **??** we discuss parameter estimation, and in Section **??** we discuss spatial prediction and uncertainty quantification of predictions. In Section **??** 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 **?**. Note that the following considers the spatial case only;

the extension to a spatio-temporal setting is given in Section **??**.

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 (**??**) 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 **?** 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 (**??**),

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

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 **??** 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 (**?**), **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 (**?**, 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 **??**.

2.3. 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** (**?**) 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 (**??**),

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

Under our modelling assumptions, the conditional density functions $[\boldsymbol{\eta} \mid \boldsymbol{\vartheta}]$ and $[\boldsymbol{\xi} \mid \sigma^2_\xi]$ 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 (**??**) 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 (**??**) that is Gaussian in terms of $\boldsymbol{u}$, with mean vector $\hat{\boldsymbol{u}}$ and covariance matrix $\boldsymbol{H}$. Substituting this approximation into (**??**) 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 **??**.

*Model fitting with **TMB***

Given as input a C++ template function which defines the complete-data log-likelihood function (**??**), **TMB** (**?**) 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** (**?**) for automatic differentiation, and the linear algebra libraries **Eigen** (**?**) and **Matrix** (**?**) 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 **??** 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 (**??**) 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 **??**. 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

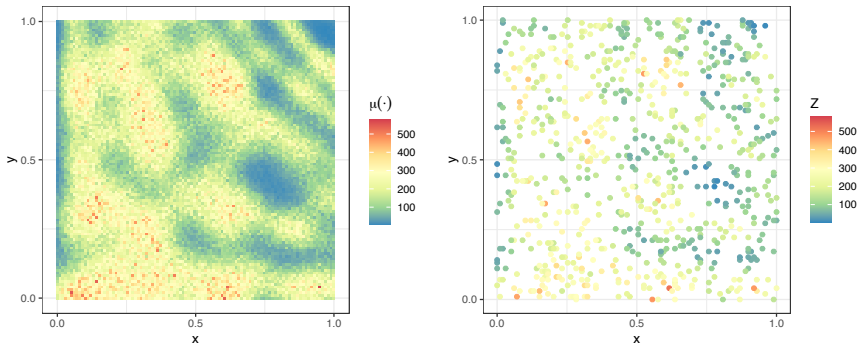

Figure 1: Simulated point-referenced spatial Poisson data set, and the true field from which the data was generated, used for the example presented in Section **??**. (Left Panel) True mean process $\mu(\cdot)$. (Right panel) Simulated observations.

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

3. Usage of new features

We now demonstrate the new features in **FRK** v.2, an overview of which is presented in Table **??**. 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 **??**. In Section **??**, we illustrate use of **FRK** v.2 using non-Gaussian spatial point-referenced data. In Section **??**, we briefly discuss extensions available in **FRK** v.2 to model data in a spatio-temporal setting. Finally, in Section **??**, 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 **??**, 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 can

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 **??**).
	`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 **??** for details on how we link the latent process $Y(\cdot)$ to the mean process $\mu(\cdot)$ when these distributions are chosen.

be performed automatically using the helper functions `auto_BAUs()` and `auto_basis()`: See **?** 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 **??**. 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`' (**?**) 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** (**?**). Figure **??** 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

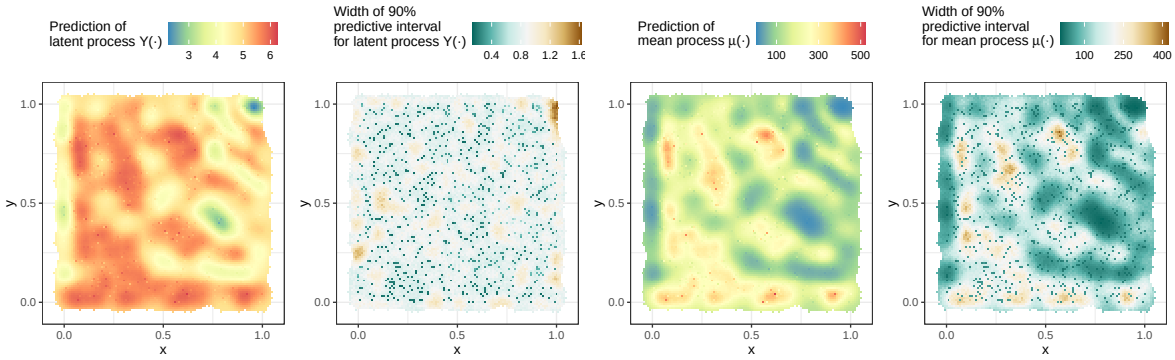

Figure 2: The prediction and prediction uncertainty quantification using the data shown in Figure **??**. (Left panel) Prediction of the latent process, $Y(\cdot)$. (Centre-left panel) Width of the 90% central predictive interval of the latent process. (Centre-right panel) Prediction of the mean process,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

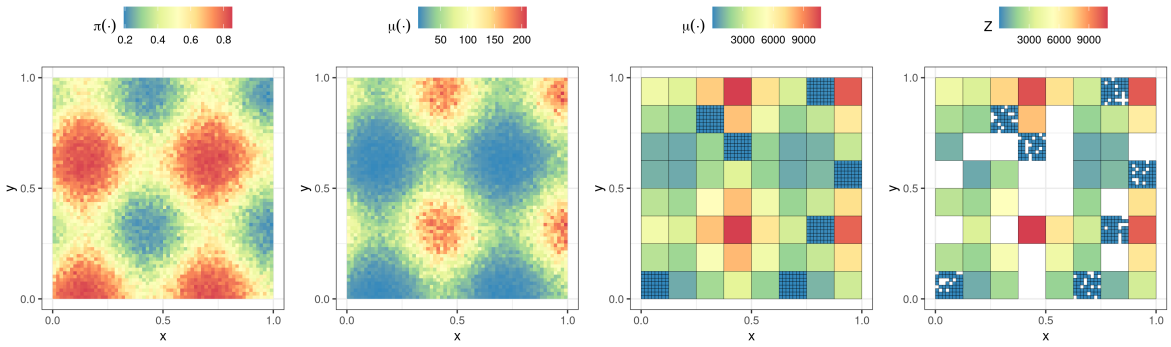

Figure 3: Simulated, areal negative-binomial data set used in the illustrative example of Section **??**. (Left panel) True probability process evaluated over the BAUs. (Centre-left panel) True mean process evaluated over the BAUs. (Centre-right panel) True mean process aggregated to the data support 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 **??** 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 **??** and the corresponding predictions. The prediction uncertainty of the mean process is roughly proportional to its prediction (as in the example of Section **??**). In contrast, the prediction

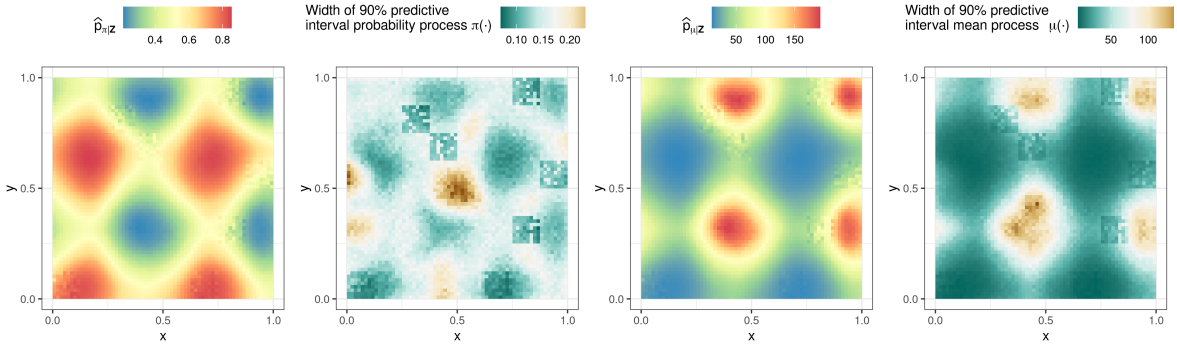

Figure 4: Prediction and prediction uncertainty quantification for the simulated negative-binomial areal toy example shown in Section **??**. (Left panel) Prediction of the probability process $\pi(\cdot)$. (Centre-left panel) Width of the 90% central predictive interval of the probability process. (Centre-right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

uncertainty of $\pi(\cdot)$ is low when the prediction is near 0 or 1, and increases when the prediction is near 0.5: 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 **??** 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 **??**.

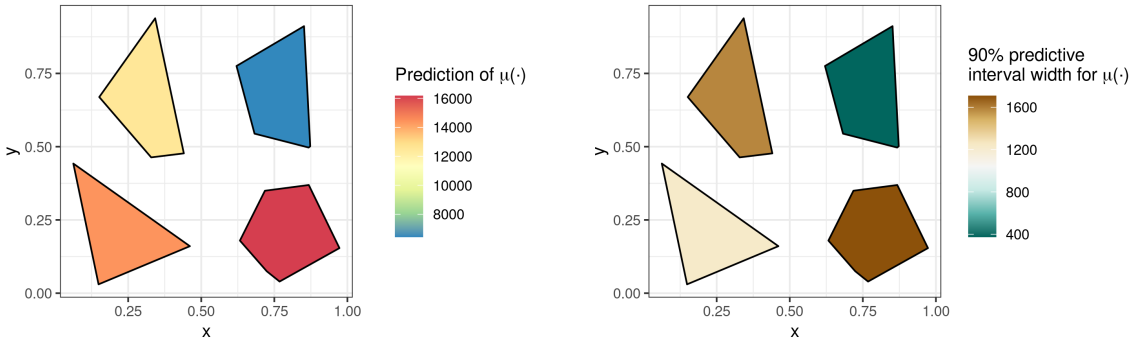

Figure 5: Prediction (left panel) and prediction uncertainty (right) of the mean process, $\mu(\cdot)$, when predicting over arbitrary polygons in the toy example of Section **??**. Note that these polygons have equal area.

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

3.4. Support for increased number of basis functions

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 **??** defines the scoring rules used throughout the remainder of this paper.

We repeated the analysis in Section **??** using one, two, and three resolutions of basis functions; Table **??** 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 (**??**).

Table 4: Numerical scoring for each competing method on the MODIS data, as presented in **?**.

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 (**?**).
[b] **TMB** supports the use of multiple cores, but this is not yet implemented in **FRK** v.2.

Although the primary motivation for **FRK** v.2 is the provision of non-Gaussian data models, 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 **?**. 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 **?** for a detailed description of the data. Table **??** replicates Table 3 of **?**, 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.