
CHALMERS
UNIVERSITY OF TECHNOLOGY

# Calculating probabilistic excursion sets and related quantities using excursions

*Journal of Statistical Software*

*August 2018, Volume 86, Issue 5.* doi: 10.18637/jss.v086.i05

# Calculating Probabilistic Excursion Sets and Related Quantities Using excursions

**David Bolin**
Chalmers University of Technology
University of Gothenburg

**Finn Lindgren**
The University of Edinburgh

**Abstract**

The R software package **excursions** contains methods for calculating probabilistic excursion sets, contour credible regions, and simultaneous confidence bands for latent Gaussian stochastic processes and fields. It also contains methods for uncertainty quantification of contour maps and computation of Gaussian integrals. This article describes the theoretical and computational methods used in the package. The main functions of the package are introduced and two examples illustrate how the package can be used.

*Keywords*: excursion sets, contour maps, contour curves, simultaneous credible bands, Gaussian integrals, R.

## 1. Introduction

The ability to find regions where a stochastic process exceeds a certain level, or is significantly different from some reference level, is important in several areas of application. Examples in geosciences include studies of air pollution (Cameletti, Lindgren, Simpson, and Rue 2013), temperature (Furrer, Knutti, Sain, Nychka, and Meehl 2007), precipitation (Sain, Furrer, and Cressie 2011), and vegetation (Eklundh and Olsson 2003; Bolin, Lindström, Eklundh, and Lindgren 2009), and similar problems can be found in a wide range of scientific fields including brain imaging (Marchini and Presanis 2003) and astrophysics (Beaky, Scherrer, and Villumsen 1992). A related problem is uncertainty quantification of contour curves and more generally of contour maps, which are often used to display estimates of continuous surfaces. The number of contours used in a contour map should typically reflect the uncertainty in the estimate, since one should be allowed to draw many contours if the uncertainty of the estimated surface is low and fewer contours if the uncertainty is high. The ability to quantify the uncertainty in the contour map is important if one should be able to choose the number of contours in a rigorous way.

The R (R Core Team 2018) software package **excursions** (Bolin and Lindgren 2018) contains functions for solving these problems for latent Gaussian models (LGMs), which is a large model class that is widely used in applications (see e.g., Rue, Martino, and Chopin 2009). The computational methods are based on the theory introduced by Bolin and Lindgren (2015, 2017) and are especially well-suited for models where the latent field has Markov properties. Solving the problems involves computing high-dimensional Gaussian integrals, which can be done more efficiently if Markov properties can be utilized. With the ability to efficiently compute Gaussian integrals, one can also compute simultaneous credible bands for latent Gaussian processes, and more generally for mixtures of Gaussian processes. This was investigated by Bolin *et al.* (2015) and **excursions** contains a slightly more general implementation of the methods from these papers.

The package supports at least three ways of specifying the model that should be analyzed. The standard method for purely Gaussian models is to specify the model by providing the parameters of the Gaussian process. For more general models, the input can either be given as Monte Carlo simulations of the process or as the result from an analysis using the R-**INLA** software package (Lindgren and Rue 2015, package available from `http://R-INLA.org/download/`). The package is available to install from the Comprehensive R Archive Network (CRAN) at `https://CRAN.R-project.org/package=excursions`. A development version of the package is available via the repository `https://bitbucket.org/davidbolin/excursions`. The development version is updated more frequently, and can easily be installed directly in R as described on the repository homepage.

The following sections describe the theoretical methods used in the package and provide an introduction to the implementation. Section 2 summarizes the theory, Section 3 introduces the main functions in the package, and Section 4 contains two examples that illustrate how the package can be used. Finally, future plans for the package is discussed in Section 5.

## 2. Definitions and computational methodology

Hierarchical models are of great importance in many areas of statistics. In the simplest form, a hierarchical model has a likelihood distribution $\pi(\mathbf{Y}|\mathbf{X},\boldsymbol{\theta})$ for observed data $\mathbf{Y}$, which is specified conditionally on a latent process of interest, $\mathbf{X}$, which has a distribution $\pi(\mathbf{X}|\boldsymbol{\theta})$. For Bayesian hierarchical models, one also specifies prior distributions for the model parameters $\boldsymbol{\theta}$. The most important special case of these models are the LGMs, which are obtained by assuming that $\mathbf{X}|\boldsymbol{\theta}$ has a Gaussian distribution. Numerous applications can be studied using models of this form, and these are therefore the main focus of the methods in **excursions**.

A statistical analysis using an LGM often concludes with reporting the posterior mean $\mathsf{E}(\mathbf{X}|\mathbf{Y})$ as a point estimate of the latent field, possibly together with posterior variances as a measure of uncertainty. In many applications, however, reporting posterior means and variances is not sufficient. As stated in the introduction, one may be interested in computing regions where the latent field exceeds some given threshold, contour curves with their associated uncertainty, or simultaneous confidence bands. In some applications, only a contour map of the posterior mean is reported, where the number of levels in the contour map should represent the uncertainty in the estimate. These are quantities that can be computed with **excursions** and we now define these in more detail before outlining how they can be computed. For details we refer to Bolin and Lindgren (2015, 2017).

## 2.1. Definitions

The main quantities that can be computed using **excursions** are (1) excursion sets, (2) contour credible regions and level avoiding sets, (3) excursion functions, (4) contour maps and their quality measures, (5) simultaneous confidence bands. This section defines these in more detail.

Throughout the section, $X(\mathbf{s})$ will denote a stochastic process defined on some domain of interest, $\Omega$, which we assume is open with a well-defined area $|\Omega| < \infty$. Since it is not necessary for the definitions, we will not explicitly state the dependency on the data, but simply allow $X$ to have some possible non-stationary distribution. In practice, however, the distribution of $X$ will typically be a posterior distribution conditionally on data, $X(\mathbf{s})|\mathbf{Y}$. For frequentist models, the distribution of $X$ could also be conditionally on for example a maximum likelihood estimate of the parameters, $X(\mathbf{s})|\mathbf{Y}, \widehat{\boldsymbol{\theta}}$.

*Excursion sets*

An excursion set is a set where the process $X(\mathbf{s})$ exceeds or goes below some given level of interest, $u$. A set where $X(\mathbf{s}) > u$ is referred to as a positive excursion set, whereas a set where $X(\mathbf{s}) < u$ is referred to as a negative excursion set. If $X(\mathbf{s}) = f(\mathbf{s})$ is a known function, these sets can be computed directly as $A_u^+(f) = \{\mathbf{s} \in \Omega; f(\mathbf{s}) > u\}$ and $A_u^-(f) = \{\mathbf{s} \in \Omega; f(\mathbf{s}) < u\}$ respectively. If $X(\mathbf{s})$ is a latent random process, one can only provide a region where it with some (high) probability exceeds the level. More specifically, the positive level $u$ excursion set with probability $\alpha$, $E_{u,\alpha}^+(X)$, is defined as the largest set so that with probability $1-\alpha$ the level $u$ is exceeded at all locations in the set,

$$E_{u,\alpha}^+ = \arg\max_D \{|D| : \mathsf{P}[D \subset A_u^+(X)] \geq 1-\alpha\}. \tag{1}$$

Similarly, the negative $u$ excursion set with probability $\alpha$, $E_{u,\alpha}^-(X)$, is defined as the largest set so that with probability $1-\alpha$ the process is below the level $u$ at all locations in the set. This set is obtained by replacing $A_u^+(X)$ with $A_u^-(X)$ in (1).

*Contour credible regions and level avoiding sets*

For a function $f$, a contour curve (or set in general) of a level $u$ is defined as the set of all level $u$ crossings. Formally, the level curve is defined as $A_u^c(f) = \left(A_u^+(f)^o \cup A_u^-(f)^o\right)^c$, where $B^o$ denotes the interior of the set $B$ and $B^c$ denotes the complement. Note that $A_u^c(f)$ not only includes the set of locations where $f(\mathbf{s}) = u$, but also all discontinuous level crossings.

For a latent random process $X$, one can only provide a credible region for the contour curve. A level $u$ contour credibility region, $E_{u,\alpha}^c(X)$, is defined as the smallest set such that with probability $1-\alpha$ *all* level $u$ crossings of $X$ are in the set. This set can be seen as the complement of the level $u$ contour avoiding set $E_{u,\alpha}(X)$, which is defined as the largest union $M_{u,\alpha}^+ \cup M_{u,\alpha}^-$, where jointly $X(\mathbf{s}) > u$ in $M_{u,\alpha}^+$ and $X(\mathbf{s}) < u$ in $M_{u,\alpha}^-$. Formally,

$$(M_{u,\alpha}^+(X), M_{u,\alpha}^-(X)) = \arg\max_{(D^+,D^-)} \{|D^- \cup D^+| : \mathsf{P}(D^- \subseteq A_u^-(X),\, D^+ \subseteq A_u^+(X)) \geq 1-\alpha\},$$

where the sets $(D^+, D^-)$ are open. The sets $M_{u,\alpha}^+$ and $M_{u,\alpha}^-$ are denoted as the pair of level $u$ avoiding sets. The notion of level avoiding sets can naturally be extended to multiple levels $u_1 < u_2 < \cdots < u_k$, which is needed when studying contour maps. In this case, the multilevel

contour avoiding set is denoted $C_{\mathbf{u},\alpha}(X)$ (For a formal definition, see Bolin and Lindgren 2017).

*Excursion functions*

Bolin and Lindgren (2015) introduced excursion functions as a tool for visualizing excursion sets simultaneously for all values of $\alpha$. For a level $u$, the positive and negative excursion functions are defined as $F_u^+(\mathbf{s}) = 1 - \inf\{\alpha; \mathbf{s} \in E_{u,\alpha}^+\}$ and $F_u^-(\mathbf{s}) = 1 - \inf\{\alpha; \mathbf{s} \in E_{u,\alpha}^-\}$, respectively. Similarly, the contour avoidance function, and the contour function are defined as $F_u(\mathbf{s}) = 1 - \inf\{\alpha; \mathbf{s} \in E_{u,\alpha}\}$ and $F_u^c(\mathbf{s}) = \sup\{\alpha; \mathbf{s} \in E_{u,\alpha}^c\}$, respectively. Finally, for levels $u_1 < u_2 < \cdots < u_k$, one can define a contour map function as $F(\mathbf{s}) = \sup\{1 - \alpha; \mathbf{s} \in C_{\mathbf{u},\alpha}\}$.

These functions take values between zero and one and each set $E_{u,\alpha}^\bullet$ can be retrieved as the $1 - \alpha$ excursion set of the function $F_u^\bullet(\mathbf{s})$. An example of an excursion set and the corresponding excursion function is shown in Figure 2.

*Contour maps and their quality measures*

For a function $f(\mathbf{s})$, a contour map $C_f$ with contour levels $u_1 < u_2 < \ldots < u_K$ is defined as the collection of contour curves $A_{u_1}^c(f), \ldots, A_{u_K}^c(f)$ and associated level sets $G_k = \{\mathbf{s} : u_k < f(\mathbf{s}) < u_{k+1}\}$, for $0 \leq k \leq K$, where one defines $u_0 = -\infty$ and $u_{K+1} = \infty$. In practice, a process $X(\mathbf{s})$ is often visualized using a contour map of the posterior mean $\mathsf{E}(X(\mathbf{s})|\mathbf{Y})$. The contour map is visualized either by just drawing the contour curves labeled by their values, or by also visualizing each level set in a specific color. The color for a set $G_k$ is typically chosen as the color that corresponds to the level $u_k^e = (u_k + u_{k+1})/2$ in a given color map. An example of this is shown in Figure 2.

In order to choose an appropriate number of contours, one must be able to quantify the uncertainty of contour maps. The uncertainty can be represented using a contour map quality measure $P$, which is a function that takes values in $[0, 1]$. Here, $P$ should be chosen in such a way that $P \approx 1$ indicates that the contour map, in some sense, is appropriate as a description of the distribution of the random field, whereas $P \approx 0$ should indicate that the contour map is inappropriate.

An example of a contour map quality measure is the normalized integral of the contour map function

$$P_0(X, C_f) = \frac{1}{|\Omega|} \int_\Omega F(\mathbf{s}) d\mathbf{s}. \tag{2}$$

The most useful quality measure is denoted $P_2$ and is defined as the simultaneous probability for the level crossings of $(u_1^e, \ldots, u_K^e)$ all falling within their respective level sets $(G_1, \ldots, G_K)$ (for details, see Bolin and Lindgren 2017).

An intuitively interpretable approach for choosing the number of contours in a contour map is to find the largest K such that $P_2$ is above some threshold. For a joint credibility of 90%, say, choose the largest number of contours such that $P_2 \geq 0.9$. How this can be done using **excursions** is illustrated in Section 4.2.

*Simultaneous confidence bands*

Especially for time series applications, the uncertainty in the latent process is often visualized using pointwise confidence bands. A pointwise confidence interval for $X$ at some location $\mathbf{s}$ is

given by $[q_{\alpha/2}(\mathbf{s}), q_{1-\alpha/2}(\mathbf{s})]$, where $q_\alpha(\mathbf{s})$ denotes the $\alpha$-quantile in the marginal distribution of $X(\mathbf{s})$.

A problem with using pointwise confidence bands is that there is not joint interpretation, and one is therefore often of interested in computing simultaneous confidence bands. For a process $X(\mathbf{s}), \mathbf{s} \in \Omega$, we define a simultaneous confidence band as the region $\{(\mathbf{s}, y) : \mathbf{s} \in \Omega, q_\rho(\mathbf{s}) \leq y \leq q_{1-\rho}(\mathbf{s})\}$. Here $\rho$ is chosen such that $\mathsf{P}(q_\rho(\mathbf{s}) < X(\mathbf{s}) < q_{1-\rho}(\mathbf{s}), \mathbf{s} \in \Omega) = 1 - \alpha$. Thus $\alpha$ controls the probability that the process is inside the confidence band at all locations in $\Omega$. An example of pointwise and simultaneous confidence bands is given in Figure 3.

## 2.2. Computational methods

If the latent process $X(\mathbf{s})$ is defined on a continuous domain $\Omega$, one has to use a discretization, $\mathbf{x}$, of the process in the statistical analysis. The vector $\mathbf{x}$ may be the process evaluated at some locations of interest or more generally the weights in a basis expansion $X(\mathbf{s}) = \sum_i \varphi_i(\mathbf{s}) x_i$. Computations in the package are in a first stage performed using the distribution of $\mathbf{x}$. If $\Omega$ is continuous, computations in a second stage interpolates the results for $\mathbf{x}$ to $\Omega$. In this section, we briefly outline the computational methods used in these two stages. As most quantities of interest can be obtained using some excursion function, we focus on how these are computed. As a representative example, we outline how $F_u^+(\mathbf{s})$ is computed in the following sections.

As before, let $\mathbf{Y}$ and $\boldsymbol{\theta}$ be vectors respectively containing observations and model parameters. Computing an excursion function $\mathbf{F}_u^+ = \{F_u^+(x_1), \ldots, F_u^+(x_n)\}$ requires computing integrals of the posterior distribution for $\mathbf{x}$. To save computation time, it is assumed that $E_{u,\alpha_1}^+ \subset E_{u,\alpha_2}^+$ if $\alpha_1 > \alpha_2$. This means that $\mathbf{F}_u$ can be obtained by first reordering the nodes and then computing a sequential integral. The reordering is in this case obtained by sorting the marginal probabilities $\mathsf{P}(x_i > u)$ (for other options, see Bolin and Lindgren 2015). After reordering, the $i$:th element of $\mathbf{F}_u^+$ is obtained as the integral

$$\int_u^\infty \pi(\mathbf{x}_{1:i}|\mathbf{Y}) d\mathbf{x}_{1:i}. \tag{3}$$

Using sequential importance sampling as described below, the whole sequence of integrals can be obtained with the same cost as computing only one integral with $i = n$, making the computation of $\mathbf{F}_u^+$ feasible also for large problems.

*Gaussian integrals*

The basis for the computational methods in the package is the ability to compute the required integral when the posterior distribution is Gaussian. In this case, one should compute an integral

$$\frac{|\mathbf{Q}|^{1/2}}{(2\pi)^{d/2}} \int_{\mathbf{u}-\boldsymbol{\mu} \leq \mathbf{x}} \exp\left(-\frac{1}{2}\mathbf{x}^\top \mathbf{Q}\mathbf{x}\right) \mathrm{d}\mathbf{x}. \tag{4}$$

Here $\boldsymbol{\mu}$ and $\mathbf{Q}$ are the posterior mean and posterior precision matrix respectively. To take advantage of the possible sparsity of $\mathbf{Q}$ if a Markovian model is used, the integral is rewritten as

$$\int_{a_d}^\infty \pi(x_d) \int_{a_{d-1}}^\infty \pi(x_{d-1}|x_d) \cdots \int_{a_2}^\infty \pi(x_2|\mathbf{x}_{3:d}) \int_{a_1}^\infty \pi(x_1|\mathbf{x}_{2:d}) \, \mathrm{d}\mathbf{x} \tag{5}$$

where, if the problem has a Markov structure, $x_i|\mathbf{x}_{i+1:d}$ only depends on a few of the elements in $\mathbf{x}_{i+1:d}$ given by the Markov structure. The integral is calculated using a sequential im-

portance sampler by starting with the integral of the last component $\pi(x_d)$ and then moving backward in the indices, see Bolin and Lindgren (2015) for further details.

*Handling non-Gaussian data*

Using the sequential importance sampler above, $\mathbf{F}_u^+$ can be computed for Gaussian models with known parameters. For more complicated models, the latent Gaussian structure has to be handled, and this can be done in different ways depending on the accuracy that is needed. **excursions** currently supports the following five methods described in detail in Bolin and Lindgren (2015): Empirical Bayes (`EB`), quantile correction (`QC`), numerical integration (`NI`), numerical integration with quantile corrections (`NIQC`), and improved numerical integration with quantile corrections (`iNIQC`).

The `EB` method is the simplest method and is based on using a Gaussian approximation of the posterior, $\pi(\mathbf{x}|\mathbf{Y}) \approx \pi_G(\mathbf{x}|\mathbf{Y}, \widehat{\boldsymbol{\theta}})$. The `QC` method uses the same Gaussian approximation but modifies the limits in the integral to improve the accuracy. The three other methods are intended for Bayesian models, where the posterior is obtained by integrating over the parameters,

$$\pi(\mathbf{x} \mid \mathbf{Y}) = \int \pi(\mathbf{x} \mid \mathbf{Y}, \boldsymbol{\theta}) \pi(\boldsymbol{\theta} \mid \mathbf{Y}) d\boldsymbol{\theta}.$$

The `NI` method approximates the integration with respect to the parameters as in the INLA method, using a sum of representative parameter configurations, and the `NIQC` and `iNIQC` methods combines this with the `QC` method to improve the accuracy further.

In general, `EB` and `QC` are suitable for frequentist models and for Bayesian models where the posterior distribution conditionally on the parameters is approximately Gaussian. The methods are equivalent if the posterior is Gaussian and in other cases `QC` is more accurate than `EB`. For Bayesian models, the `NI` method is, in general, more accurate than the `QC` method, and for non-Gaussian likelihoods, the `NIQC` and `iNIQC` methods can be used for improved results. In general the accuracy and computational cost of the methods are as follows:

$$\text{Accuracy: } \texttt{EB} < \texttt{QC} < \texttt{NI} < \texttt{NIQC} < \texttt{iNIQC}.$$
$$\text{Computational cost: } \texttt{EB} \approx \texttt{QC} < \texttt{NI} \approx \texttt{NIQC} < \texttt{iNIQC}.$$

If the main purpose of the analysis is to construct excursion or contour sets for low values of $\alpha$, we recommend using the `QC` method for problems with Gaussian likelihoods and the `NIQC` method for problems with non-Gaussian likelihoods. The increase in accuracy of the `iNIQC` method is often small compared to the added computational cost.

*Continuous domain interpolations*

For a continuous spatial domain, the excursion function $F_u^+(\mathbf{s})$ can be approximated using $\mathbf{F}_u^+$ computed at discrete point locations. The main idea is to interpolate $\mathbf{F}_u^+$ assuming monotonicity of the random field between the discrete computation locations. Specifically, assume that the values of $\mathbf{F}_u^+$ correspond to the values at the vertices of some triangulated mesh such as the one shown in the left panel of Figure 2. If the excursion set $E_{u,\alpha}^+(X)$ should be computed for some specific value of $\alpha$, one has to find the $1-\alpha$ contour for $F_u^+(\mathbf{s})$. For interpolation to a location $\mathbf{s}$ within a specific triangle $\mathcal{T}$ with corners in $\mathbf{s}_1, \mathbf{s}_2$, and $\mathbf{s}_3$, **excursions** by default uses log-linear interpolation, $F_u(\mathbf{s}) = \exp\{\sum_{k=1}^3 w_k \log[F_u(\mathbf{s}_k)]\}$. Here

$\{(w_1, w_2, w_3);\ w_1, w_2, w_3 \geq 0,\ \sum_{k=1}^{3} w_i = 1\}$ are the barycentric coordinates of $\mathbf{s}$ within the triangle.

Further technical details of the continuous domain construction are given in Bolin and Lindgren (2017). Studies of the resulting continuous domain excursion sets in Bolin and Lindgren (2017) indicate that the log-linear interpolation method results in sets with coverage probability on target or slightly above target for large target probabilities. An example of a continuous domain excursion set for a triangulated mesh is shown in the middle panel of Figure 2. In the right panel of the figure, the interpolated function $F_u^+(\mathbf{s})$ is shown. The code that generates the figure is explained in the next section.

## 3. Implementation

The functions in **excursions** can be divided into four main categories depending on what they compute: (1) Excursion sets and credible regions for contour curves, (2) Quality measures for contour maps, (3) Simultaneous confidence bands, and (4) Utility such as Gaussian integrals and continuous domain mappings. The main functions come in at least three different versions taking different input: (1) The parameters of a Gaussian process, (2) Results from an analysis using the R-**INLA** software package, and (3) Monte Carlo simulations of the process. These different categories are described in further detail below.

Much of the computations in the package is done in C functions. These functions use methods from a number of C and Fortran libraries, such as **BLAS** (Dongarra, Du Croz, Hammarling, and Duff 1990), **LAPACK** (Anderson *et al.* 1999), and **CHOLMOD** (Chen, Davis, Hager, and Rajamanickam 2008) for efficient matrix manipulations together with function from the GNU Scientific library (Galassi and Gough 2006) and several different reordering methods. Notably, the **CAMD** library (Amestoy, Davis, and Duff 1996, 2004) is used for constrained approximate minimum degree orderings.

As an example that will be used to illustrate the methods in later sections, we generate data $Y_i \sim \mathsf{N}(X(\mathbf{s}_i), \sigma^2)$ at some locations $\mathbf{s}_1, \ldots, \mathbf{s}_{100}$ where $X(\mathbf{s})$ is a Gaussian random field specified using a stationary SPDE model (Lindgren, Rue, and Lindström 2011).

```
R> x <- seq(from = 0, to = 10, length.out = 20)
R> mesh <- inla.mesh.create(lattice = inla.mesh.lattice(x = x, y = x),
+    extend = FALSE, refine = FALSE)
R> spde <- inla.spde2.matern(mesh, alpha = 2)
R> obs.loc <- 10 * cbind(runif(100), runif(100))
R> Q <- inla.spde2.precision(spde, theta = c(log(sqrt(0.5)), 0))
R> x <- inla.qsample(Q = Q, seed = seed)
```

Based on the observations, we calculate the posterior distribution of the latent field, which is Gaussian with mean `mu.post` and precision matrix `Q.post`, these are computed as follows. We refer to Lindgren and Rue (2015) for details about the R-**INLA** related details in the code.

```
R> A <- inla.spde.make.A(mesh = mesh, loc = obs.loc)
R> sigma2.e = 0.01
R> Y <- as.vector(A %*% x + rnorm(100) * sqrt(sigma2.e))
R> Q.post <- (Q + (t(A) %*% A) / sigma2.e)
R> mu.post <- as.vector(solve(Q.post, (t(A) %*% Y) / sigma2.e))
```

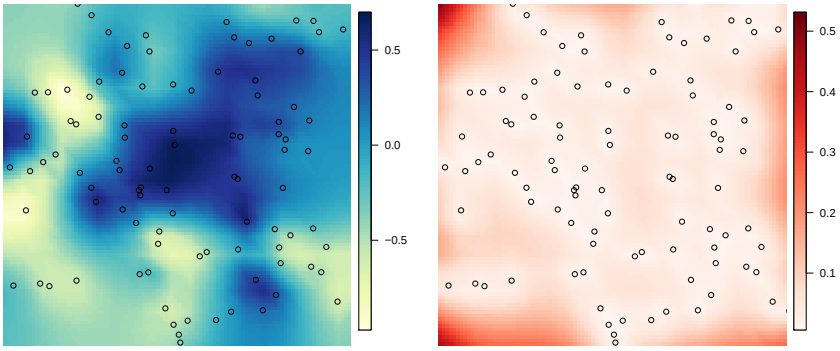


Figure 1: Posterior mean (left) and posterior standard deviations of an example using simulated data. The measurement locations are marked with circles in both panels.

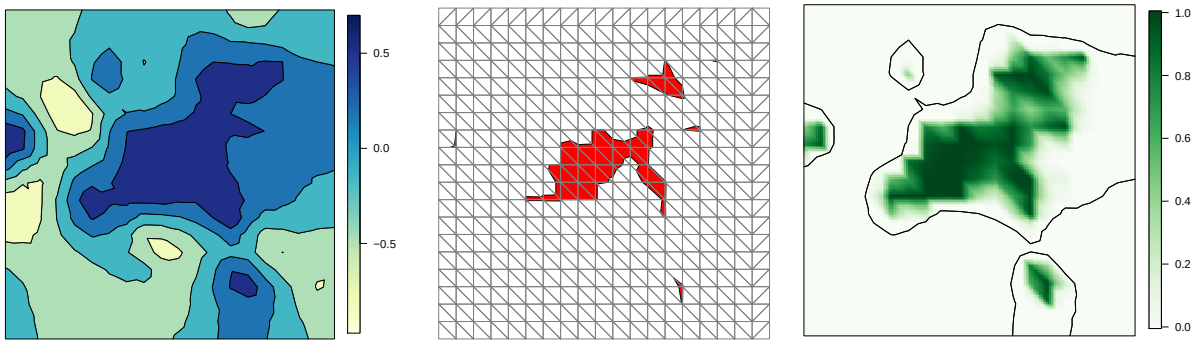


Figure 2: Contour map of the posterior mean shown in Figure 1 (left), a triangulated mesh and interpolated excursion set $E^+_{0,0.1}$ (mid), and the interpolated excursion function $F^+_0(s)$ (right). In the right panel, the zero-level contour is also plotted.

Figure 1 shows the posterior mean and the posterior standard deviations. A contour map of the posterior mean is shown in Figure 2.

### 3.1. Excursion sets and contour credible regions

The main function for computing excursion sets and contour credible regions is `excursions`. A typical call to the function looks like

```
R> res.exc <-  excursions(mu = mu.post, Q = Q.post, alpha = 0.1, type = ">",
+    u = 0, F.limit = 1)
```

Here, `mu` and `Q` are the mean vector and precision matrix for the joint distribution of the Gaussian vector `x`. The arguments `alpha`, `u`, and `type` are used to specify what type of excursion set that should be computed. `alpha` is the error probability, `u` is the excursion or contour level, and `type` determines what type of region that is considered: '>' for positive excursion regions, '<' for negative excursion regions, '!=' for contour avoiding regions, and '=' for contour credibility regions. Thus, the call above computes the excursion set $E^+_{0,0.1}$.

The argument `F.limit` is used to specify when to stop the computation of the excursion function. In this case with `F.limit=1`, all values of $\mathbf{F}_u^+$ are computed, but the computation time can be reduced by decreasing the value of `F.limit`.

The function has the EB method as default strategy for handling the possible latent Gaussian structure. In the simulated example, the likelihood is Gaussian and the parameters are assumed to be known, so the EB method is exact. The QC method can be used instead by specifying `method="QC"`. In this case, the argument `rho` should be used to also provide a vector with point-wise marginal probabilities: $P(x_i > u)$ for positive excursions and contour regions, and $P(x_i < u)$ for negative excursions. In the situation when $\pi(\mathbf{x}|\mathbf{Y}, \boldsymbol{\theta})$ is Gaussian but $\pi(\mathbf{x}|\mathbf{Y})$ is not, the marginal probabilities should be calculated under the distribution $\pi(\mathbf{x}|\mathbf{Y})$ and `mu` and `Q` should be chosen as the mean and precision for the distribution $\pi(\mathbf{x}|\mathbf{Y}, \hat{\boldsymbol{\theta}})$ where $\hat{\boldsymbol{\theta}}$ is the MAP or ML estimate of the parameters.

### *The INLA interface*

The function `excursions.inla` is used to compute excursion sets and credible regions for contour curves for latent Gaussian models that have been estimated using R-**INLA**. It takes the same arguments as `excursions`, except that `mu` and `Q` are replaced with arguments related to R-**INLA**. A basic call to the function `excursions.inla` looks like

```
R> excursions.inla(result.inla, name, alpha, u, type)
```

Here `result.inla` is the output from a call to the `inla` function, and `name` is the name of the component in the output that the computations should be done for. For more complicated models, one typically specifies the model in R-**INLA** using an `inla.stack` object. In this case, the call to `excursions.inla` will instead look like

```
R> excursions.inla(result.inla, stack, tag, alpha, u, type)
```

Here `stack` is now the stack object and `tag` is the name of the component in the stack for which the computations should be done. The typical usage of the `tag` argument is to have one part of the stack that contains the locations where measurements are taken, and another that contains the locations where the output should be computed.

`excursions.inla` has support for all strategies described in Section 2 for handling latent Gaussian structures: The argument `method` can be one of `"EB"`, `"QC"`, `"NI"`, `"NIQC"`, and `"iNIQC"`.

### *Analysis of Monte Carlo samples*

The function `excursions.mc` can be used to post-process Monte Carlo model simulations in order to compute excursion sets and credible regions. For this function, the model is not specified explicitly. Instead a $d \times N$ matrix `X` containing $N$ Monte Carlo simulations of the $d$ dimensional process of interest is provided. A basic call to the function looks like

```
R> excursions.mc(X, u, type)
```

where `u` again determines the level of interest and `type` defines the type of set that should be computed. It is important to note that this function does all computations purely based

on the Monte Carlo samples that are provided, and it does not use any of the computational methods based on sequential importance sampling for Gaussian integrals that the is the basis for the previous methods. This means that this function in one sense is more general as `X` can be samples from any model, not necessarily a latent Gaussian model. The price that has to be paid for this generality is that the only way of increasing the accuracy of the results is to increase the number of Monte Carlo samples that are provided to the function.

### 3.2. Analysis of contour maps

The main function for analysis of contour maps is `contourmap`. A basic call to the function looks like

```
R> res.con <- contourmap(mu = mu.post, Q = Q.post, n.levels = 4,
+    alpha = 0.1, compute = list(F = TRUE, measures = c("PO")))
```

Here, `mu` is again the mean value and `Q` is the precision matrix of the model. The parameter `n.levels` sets the number of contours that should be used in the contour map, and these are spaced equidistant in the range of `mu` by default. Other types of contour maps can be obtained using the `type` argument. For manual specification of the levels, the `levels` argument can be used instead. By default, the function computes the specified contour map but no quality measures and it does not compute the contour map function. If quality measures should be computed, this is specified using the `compute` argument. This argument is also used to decide whether the contour map function $F$ should be computed.

As for `excursions`, this function comes in two other versions depending on the form of the input: `contourmap.inla` for model specification using an R-**INLA** object, or `contourmap.mc` for model specification using Monte Carlo simulations of the model. The model specification for these functions is identical to that in the corresponding `excursions` functions.

### 3.3. Continuous domain interpretations

A common scenario is that the input used in `contourmap` or `excursions` represents the value of the model at some discrete locations in a continuous domain of interest. In this case, the function `continuous` can be used to interpolate the discretely computed values by assuming monotonicity of the random field in between the discrete computation locations, as discussed in Section 2. A typical call to the function looks like

```
R> sets.exc <- continuous(ex = res.exc, geometry = mesh, alpha = 0.1)
```

Here `ex` is the result of the call to `contourmap` or `excursions` and `alpha` is the error probability of interest for the excursion set or credible region computation. The argument `geometry` specifies the geometric configuration of the values in input `ex`, either as a general triangulation geometry or as a lattice. A lattice can be specified as an object of the form `list(x, y)` where `x` and `y` are vectors, or as `list(loc, dims)` where `loc` is a two-column matrix of coordinates, and `dims` is the lattice size vector. If R-**INLA** is used, the lattice can also be specified as an `inla.mesh.lattice` object. In all cases, the input is treated topologically as a lattice with lattice boxes that are assumed convex. A triangulation geometry is specified as an `inla.mesh` object. Finally, an argument `output` can be used to specify what type of object should be

generated. The options are currently `sp` which gives a `SpatialPolygons` (Bivand, Pebesma, and Gómez-Rubio 2013) object, and `inla` which gives an `inla.mesh.segment` object.

### 3.4. Simultaneous confidence bands

The function `simconf` can be used for calculating simultaneous confidence bands for a Gaussian process $X(s)$. A basic call to the function looks like

```
R> simconf(alpha, mu, Q)
```

where `alpha` is the error (1 - coverage) probability, `mu` is the mean value vector for the process, and `Q` is the precision matrix for the process. The function has a few optional arguments similar to those of `excursions`, all listed in the documentation of the function. The function returns upper and lower limits for both pointwise and simultaneous confidence bands.

As for `excursions` and `contourmap`, there is also a version of `simconf` that can be used to analyze R-**INLA** models (`simconf.inla`) and a version that can analyze Monte Carlo samples (`simconf.mc`). Furthermore, there is a version `simconf.mixture` which computes simultaneous confidence regions for Gaussian mixture models with a joint distribution on the form

$$\pi(x) = \sum_{k=1}^{K} w_k \mathsf{N}(\mu_k, Q_k^{-1}).$$

This particular function was used to analyze the models in Bolin *et al.* (2015) and Guttorp *et al.* (2014), but is also used internally by `simconf.inla`.

### 3.5. Gaussian integrals

Among the utility functions in the package, the function `gaussint` can be especially useful also in a larger context. It contains the implementation of the sequential importance sampling method for computing Gaussian integrals, described in Section 2. This function has two features that separates it from many other functions for computing Gaussian integrals: Firstly it is based on the precision matrix of the Gaussian distribution, and sparsity of this matrix can be utilized to decrease computation time. Secondly, the integration can be stopped as soon as the value of the integral in the sequential integration goes below some given value $1 - \alpha$. If one only is interested in the exact value of the integral given that it is larger than some value $1 - \alpha$, this option can save a lot of computation time.

A basic call to the function looks like

```
R> gaussint(mu, Q, a, b)
```

where `mu` is the mean value vector, `Q` is the precision matrix, `a` is a vector of the lower limits in the integral, and `b` contains the upper integration limits. If the Cholesky factor of `Q` is known beforehand, this can be supplied to the function using the `Q.chol` argument. An argument `alpha` is used to set the computational $1 - \alpha$ limit for the integral. The function returns an estimate of the integral as well as an error estimate. If the error estimate is too high, the precision can be increased by increasing the `n.iter` argument of the function.

### 3.6. Plotting

The **excursions** package contains various functions that are useful for visualization. The function `tricontourmap` can be used for visualization of contour maps computed on triangulated meshes. The following code plots the posterior mean using the contour map we previously computed.

```
R> set.sc <- tricontourmap(mesh, z = mu.post, levels = res.con$u)
R> cmap <- colorRampPalette(brewer.pal(9, "YlGnBu"))(100)
R> cols <- contourmap.colors(res.con, col = cmap)
R> plot(set.sc$map, col = cols)
```

Here `contourmap.colors` is used to find appropriate colors for each set in the contour map, based on the color map `cmap` that was defined using the **RColorBrewer** (Neuwirth 2014) package. The results of the following commands are shown in Figure 2. The estimated excursion set $E^+_{u,\alpha}(X)$, can be visualized as

```
R> plot(sets.exc$M["1"], col = "red", xlim = range(mesh$loc[,1]),
+    ylim = range(mesh$loc[,2]))
R> plot(mesh, vertex.color = rgb(0.5, 0.5, 0.5), draw.segments = FALSE,
+    edge.color = rgb(0.5, 0.5, 0.5), add = TRUE)
```

The second `plot` command adds the mesh to the plot so that we can see how the set is interpolated by the `continuous` function. Finally, the interpolated excursion function $F^+_u(\mathbf{s})$, can be plotted easily using the `inla.mesh.projector` function from the R-**INLA** package.

```
R> cmap.F <- colorRampPalette(brewer.pal(9, "Greens"))(100)
R> proj <- inla.mesh.projector(sets.exc$F.geometry, dims = c(200, 200))
R> image(proj$x, proj$y, inla.mesh.project(proj, field = sets.exc$F),
+    col = cmap.F, axes = FALSE, xlab = "", ylab = "", asp = 1)
R> con <- tricontourmap(mesh, z = mu.post, levels = 0)
R> plot(con$map, add = TRUE)
```

The final two lines computes the level zero contour curve and plots it in the same figure as the interpolated excursion function. The colorbars in the figures are plotted using the **fields** (Nychka, Furrer, Paige, and Sain 2018) package.

## 4. Two applications

This section presents two examples that illustrate how **excursions** can be used.

### 4.1. Time series data: Tokyo rainfall

To illustrate the methods in the package, we use the much analyzed binomial time series from Kitagawa (1987). Each day during the years 1983 and 1984, it was recorded whether there was more than 1 mm rainfall in Tokyo. Of interest is to study the underlying probability of rainfall as a function of day of the year. The data is modelled as $y_i \sim Bin(n_i, p_i)$ for calendar day $i = 1, \ldots, 366$. Here $n_i = 2$ for all days except for February 29 ($i = 60$) which only

occurred during the leap year of 1984. The probability $p_i$ is modeled as a logit-transformed Gaussian process.

The model and the following R-**INLA** implementation of the model is described further in Lindgren and Rue (2015).

```
R> data("Tokyo")
R> mesh <- inla.mesh.1d(seq(1, 367, length = 25), interval = c(1, 367),
+    degree = 2, boundary = "cyclic")
R> kappa <- 1e-3
R> tau <- 1 / (4 * kappa^3)^0.5
R> spde <- inla.spde2.matern(mesh, constr = FALSE, theta.prior.prec = 1e-4,
+    B.tau = cbind(log(tau), 1), B.kappa = cbind(log(kappa), 0))
R> A <- inla.spde.make.A(mesh, loc = Tokyo$time)
R> time.index <- inla.spde.make.index("time", n.spde = spde$n.spde)
R> stack <- inla.stack(data = list(y = Tokyo$y, link = 1, Ntrials = Tokyo$n),
+    A = list(A), effects = list(time.index), tag = "est")
R> formula <- y ~ -1 + f(time, model = spde)
R> data <- inla.stack.data(stack)
```

Next, the model is estimated using the `inla` function. Since we want to analyze the output using **excursions**, the additional option `control.compute = list(config = TRUE)` must be specified in the `inla` function. This makes the function save some extra output needed by **excursions**.

```
R> result <- inla(formula, family = "binomial", data = data,
+    Ntrials = data$Ntrials, control.predictor = list(
+    A = inla.stack.A(stack), link = data$link, compute = TRUE),
+    control.compute = list(config = TRUE))
```

We now have estimates of the posterior mean and marginal confidence intervals for the probability of rain for each day. However, if we also want joint confidence bands, we can estimate these using `simconf.inla` as

```
R> res <- simconf.inla(result, stack, tag = "est", alpha = 0.05, link = TRUE)
```

Note the argument `link=TRUE` which tells the function that the results should be returned in the scale of the data, and not in the scale of the linear predictor. Next, we plot the results, showing the marginal confidence bands with dashed lines and the simultaneous confidence band with dotted lines. The results are shown in Figure 3.

```
R> index <- inla.stack.index(stack, "est")$data
R> plot(Tokyo$time, Tokyo$y / Tokyo$n, xlab = "Day", ylab = "Probability")
R> lines(result$summary.fitted.values$mean[index])
R> matplot(cbind(res$a.marginal, res$b.marginal), type = "l", lty = 2,
+    col = 1, add = TRUE)
R> matplot(cbind(res$a, res$b), type = "l", lty = 3, col = 1, add = TRUE)
```

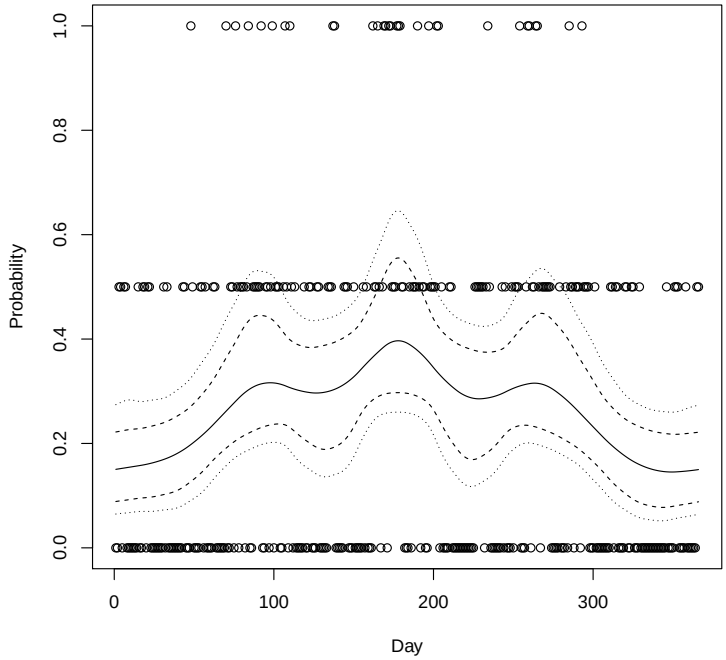


Figure 3: Empirical and model-based Binomial probability estimates for the Tokyo rainfall data set, with a marginal 95% confidence band (dashed) and a simultaneous confidence band (dotted). The empirical probability estimates are the proportion of observed rainfall days for each day of the year.

## 4.2. Spatial data: Parana precipitation

We will now illustrate how **excursions** can be used for a spatial dataset. In order to keep the parts of the code that are not relevant to **excursions** brief, we again use data available in R-**INLA**. The dataset consists of daily rainfall data for each day of the year 2011, at 616 locations in and around the state of Paraná in Brazil. In the following, we analyze the data from the month of January.

The statistical model used for the data is a latent Gaussian model, where the precipitation measurements are assumed to be Γ-distributed with a spatially varying mean. The mean is modeled as a log-Gaussian Matérn field specified as an SPDE model. Details of the model and the following INLA implementation can be found in the SPDE tutorial available on the R-**INLA** homepage, see Wallin and Bolin (2015) for an analysis of the data using a different non-Gaussian SPDE model.

We start by loading the data and defining the model:

```
R> data("PRprec")
R> data("PRborder")
R> Y <- rowMeans(PRprec[, 3 + (1:31)])
R> ind <- !is.na(Y)
R> Y <- Y[ind]
R> coords <- as.matrix(PRprec[ind, 1:2])
```

```
R> b <- inla.nonconvex.hull(coords, -0.03, -0.05, resolution = c(100, 100))
R> prmesh <- inla.mesh.2d(boundary = b, max.edge = c(0.45, 1), cutoff = 0.2)
R> A <- inla.spde.make.A(prmesh, loc = coords)
R> spde <- inla.spde2.matern(prmesh, alpha = 2)
R> mesh.index <- inla.spde.make.index(name = "field", n.spde = spde$n.spde)
```

The measurement stations are spatially irregular, but we are interested in making predictions to a regular lattice within the state. In order to do continuous domain interpretations, we define the lattice locations as a lattice object using the `submesh` function of the **excursions** package. The function `inout` from the package **splancs** (Rowlingson and Diggle 2017) is used to find the locations on the lattice that are within the region of interest.

```
R> nxy <- c(50, 50)
R> projgrid <- inla.mesh.projector(prmesh, xlim = range(PRborder[, 1]),
+    ylim = range(PRborder[, 2]), dims = nxy)
R> xy.in <- inout(projgrid$lattice$loc, cbind(PRborder[, 1], PRborder[, 2]))
R> submesh = submesh.grid(matrix(xy.in, nxy[1], nxy[2]),
+    list(loc = projgrid$lattice$loc, dims = nxy))
```

Next, we define the `stack` objects and estimate the model using `inla`. Again note that we have to set the `control.compute` argument since the result is to be used by **excursions**.

```
R> A.prd <- inla.spde.make.A(prmesh, loc = submesh$loc)
R> stk.prd <- inla.stack(data = list(y = NA), A = list(A.prd, 1),
+    effects = list(c(mesh.index, list(Intercept = 1)),
+    list(lat = submesh$loc[,2], lon = submesh$loc[,1])), tag = "prd")
R> stk.dat <- inla.stack(data = list(y = Y), A = list(A, 1),
+    effects = list(c(mesh.index, list(Intercept = 1)),
+    list(lat = coords[,2], lon = coords[,1])), tag = "est")
R> stk <- inla.stack(stk.dat, stk.prd)
R> r <- inla(y ~ -1 + Intercept + f(field, model = spde), family = "Gamma",
+    data = inla.stack.data(stk), control.compute = list(config = TRUE),
+    control.predictor = list(A = inla.stack.A(stk), compute = TRUE,
+      link = 1))
```

We now want to find areas that likely experienced large amounts of precipitation. In the following code, we compute the excursion set for the posterior mean for the level 7 mm of precipitation. To indicate that this level is in the scale of the data, and not in the scale of the linear predictor, we use the `u.link=TRUE` argument in the `excursions` call.

```
R> exc = excursions.inla(r, stk, tag = "prd", u = 7, u.link = TRUE,
+    type = ">", F.limit = 0.6, method = "QC")
R> sets <- continuous(exc, submesh, alpha = 0.1)
```

We also compute the contour curve for the level of interest on the continuous domain, using the `tricontourmap` function.

```
R> con <- tricontourmap(submesh, z = exc$mean, levels = log(7))
```

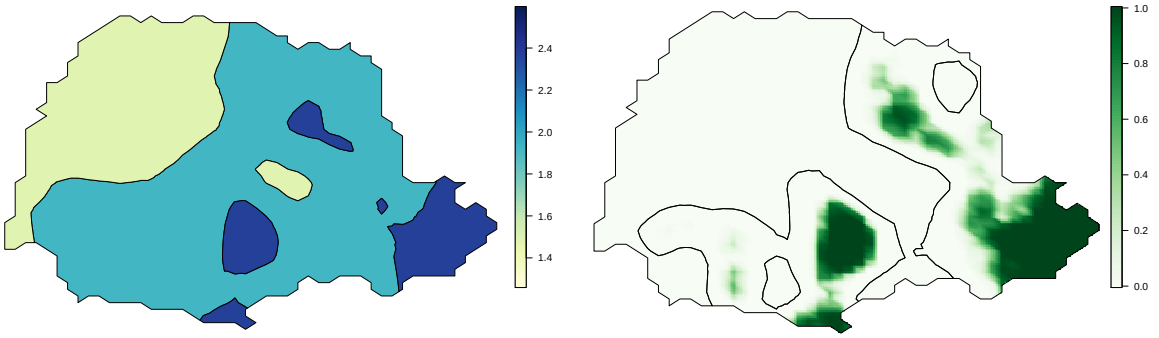


Figure 4: A contour map of the posterior mean of log precipitation (left), and the excursion function for the positive $u = \log(7)$ excursion set (right). In the right panel, also the $u = \log(7)$ contour is shown.

We plot the resulting continuous domain excursion function together with the contour curve using the following commands. The result is shown in the right panel of Figure 4.

```
R> proj <- inla.mesh.projector(sets$F.geometry, dims = c(300,200))
R> image(proj$x, proj$y, inla.mesh.project(proj, field = sets$F),
+    col = cmap.F, axes = FALSE, xlab = "", ylab = "", asp = 1)
R> plot(con$map, add = TRUE)
```

Next, the posterior mean is visualized using a contour map.

```
R> lp <- contourmap.inla(r, stack = stk, tag = "prd", n.levels = 2,
+    compute = list(F = FALSE))
R> tmap <- tricontourmap(submesh, z = lp$meta$mu, levels = lp$meta$levels)
R> plot(tmap$map, col = contourmap.colors(lp, col = cmap))
```

Here, the `contourmap.inla` computes the levels of the contour map and `tricontourmap` computes the contour map on the mesh. Finally, `contourmap.colors` is used to compute appropriate colors for visualizing the contour map. The result is shown in the left panel of Figure 4.

The contour map we computed had two contours, and a relevant question is now if this is an appropriate number. To investigate this, we compute the $P_2$ quality measure for this contour map and for contour maps with one and three levels.

```
R> lp1 <- contourmap.inla(r, stack = stk, tag = "prd", n.levels = 1,
+    compute = list(F = FALSE, measures = c("P2")))
R> lp2 <- contourmap.inla(r, stack = stk, tag = "prd", n.levels = 2,
+    compute = list(F = FALSE, measures = c("P2")), n.iter=40000)
R> lp3 <- contourmap.inla(r, stack = stk, tag = "prd", n.levels = 3,
+    compute = list(F = FALSE, measures = c("P2")))
R> cat(sprintf("%.2f %.2f %.2f",lp1$P2, lp2$P2, lp3$P2))

1.00 0.20 0.00
```

We see that using only one contour gives very high credibility, whereas using three contours give a credibility that is close to zero. The contour map with two contours has a credibility around 0.2 and seems to be a good compromise for this application.

## 5. Discussion

The **excursions** package was first developed for calculating probabilistic excursion sets and contour credible regions for latent Gaussian stochastic processes and fields. Since the early versions, the scope of the package has grown and it now contains functions for several other related computations, such as simultaneous confidence bands, uncertainty quantification of contour maps, and computations of Gaussian integrals.

Some of the functionality in **excursions** can also be found in other packages. For example, **ExceedanceTools** (French 2014) provides an alternative method for computing excursion sets and uncertainty regions for contour curves for Gaussian models. The package **mvtnorm** (Genz *et al.* 2018) contains functions for computing integrals of Gaussian distributions, but does not have support for computations based on sparse precision matrices. Finally, there are numerous packages with the ability to compute simultaneous confidence bands for various models, such as semiparametric regression models using **AdaptFitOS** (Wiesenfarth, Krivobokova, Klasen, and Sperlich 2012) or nonparametric regression models for functional data using **SCBmeanfd** (Degras 2016). Comparing the methods in **excursions** to the corresponding methods in these packages is an interesting topic for future studies.

Future work also includes further development of the package, and new features are added as they are needed. One main focus for the current development is to add functionality for large scale computations, where sparse Cholesky factorization is computationally infeasible. We also plan to add functionality for computations needed in spatial extreme value theory, where certain Gaussian integrals often are needed during likelihood inference. Finally, the technical aspects of the functions are also being improved, and future releases will introduce improvements to both the continuous domain interpretations and to the algorithms for finding the largest excursion sets.

## References

Amestoy P, Davis TA, Duff IS (1996). "An Approximate Minimum Degree Ordering Algorithm." *SIAM Journal on Matrix Analysis and Applications*, **17**(4), 886–905. doi:10.1137/s0895479894278952.

Amestoy P, Davis TA, Duff IS (2004). "Algorithm 837: AMD, an Approximate Minimum Degree Ordering Algorithm." *ACM Transactions on Mathematical Software*, **30**(3), 381–388. doi:10.1145/1024074.1024081.

Anderson E, Bai Z, Bischof C, Blackford LS, Demmel J, Dongarra J, Du Croz J, Hammarling S, Greenbaum A, McKenney A, and Sorensen D (1999). ***LAPACK*** *Users' Guide*. 3rd edition. Society for Industrial and Applied Mathematics, Philadelphia.

Beaky MM, Scherrer RJ, Villumsen JV (1992). "Topology of Large-Scale Structure in Seeded

Hot Dark Matter Models." *Astrophysical Journal, Part 1*, **387**, 443–448. doi:10.1086/171097.

Bivand RS, Pebesma E, Gómez-Rubio V (2013). *Applied Spatial Data Analysis with R*. Springer-Verlag, New York. doi:10.1007/978-1-4614-7618-4.

Bolin D, Guttorp P, Januzzi A, Jones D, Novak M, Podschwit H, Richardson L, Särkkä A, Sowder C, Zimmerman A (2015). "Statistical Prediction of Global Sea Level from Global Temperature." *Statistica Sinica*, **25**, 351–367. doi:10.5705/ss.2013.222w.

Bolin D, Lindgren F (2015). "Excursion and Contour Uncertainty Regions for Latent Gaussian Models." *Journal of the Royal Statistical Society B*, **77**, 85–106. doi:10.1111/rssb.12055.

Bolin D, Lindgren F (2017). "Quantifying the Uncertainty of Contour Maps." *Journal of Computational and Graphical Statistics*, **26**(3), 513–524. doi:10.1080/10618600.2016.1228537.

Bolin D, Lindgren F (2018). ***excursions**: Excursion Sets and Contour Credible Regions for Latent Gaussian Models*. R package version 2.4.4, URL https://CRAN.R-project.org/package=excursions.

Bolin D, Lindström J, Eklundh L, Lindgren F (2009). "Fast Estimation of Spatially Dependent Temporal Vegetation Trends Using Gaussian Markov Random Fields." *Computational Statistics & Data Analysis*, **53**, 2885–2896. doi:10.1016/j.csda.2008.09.017.

Cameletti M, Lindgren F, Simpson D, Rue H (2013). "Spatio-Temporal Modeling of Particulate Matter Concentration through the SPDE Approach." *AStA Advances in Statistical Analysis*, **97**(2), 109–131. doi:10.1007/s10182-012-0196-3.

Chen Y, Davis TA, Hager WW, Rajamanickam S (2008). "Algorithm 887: CHOLMOD, Supernodal Sparse Cholesky Factorization and Update/Downdate." *ACM Transactions on Mathematical Software*, **35**(3), 22. doi:10.1145/1391989.1391995.

Degras D (2016). ***SCBmeanfd**: Simultaneous Confidence Bands for the Mean of Functional Data*. R package version 1.2.2, URL https://CRAN.R-project.org/package=SCBmeanfd.

Dongarra JJ, Du Croz J, Hammarling S, Duff IS (1990). "A Set of Level 3 Basic Linear Algebra Subprograms." *ACM Transactions on Mathematical Software*, **16**(1), 1–17. doi:10.1145/77626.79170.

Eklundh L, Olsson L (2003). "Vegetation Index Trends for the African Sahel 1982–1999." *Geophysical Research Letters*, **30**, 1430–1433. doi:10.1029/2002gl016772.

French J (2014). ***ExceedanceTools**: Confidence Regions for Exceedance Sets and Contour Lines*. R package version 1.2.2, URL https://CRAN.R-project.org/package=ExceedanceTools.

Furrer R, Knutti R, Sain SR, Nychka DW, Meehl GA (2007). "Spatial Patterns of Probabilistic Temperature Change Projections from a Multivariate Bayesian Analysis." *Geophysical Research Letters*, **34**. doi:10.1029/2006gl027754.

Galassi M, Gough B (2006). *GNU Scientific Library: Reference Manual*. Network Theory Limited, Bristol.

Genz A, Bretz F, Miwa T, Mi X, Leisch F, Scheipl F, Hothorn T (2018). ***mvtnorm**: Multivariate Normal and t Distributions*. R package version 1.0-8, URL http://CRAN.R-project.org/package=mvtnorm.

Guttorp P, Januzzi A, Novak M, Podschwit H, Richardson L, Sowder CD, Zimmerman A, Bolin D, Särkkä A (2014). "Assessing the Uncertainty in Projecting Local Mean Sea Level from Global Temperature." *Journal of Applied Meteorology and Climatology*, **53**(9), 2163–2170. doi:10.1175/jamc-d-13-0308.1.

Kitagawa G (1987). "Non-Gaussian State-Space Modeling of Nonstationary Time Series." *Journal of the American Statistical Association*, **82**(400), 1032–1041. doi:10.1080/01621459.1987.10478534.

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

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

Marchini J, Presanis A (2003). "Comparing Methods of Analyzing fMRI Statistical Parametric Maps." *NeuroImage*, **22**, 1203–1213. doi:10.1016/j.neuroimage.2004.03.030.

Neuwirth E (2014). ***RColorBrewer**: ColorBrewer Palettes*. R package version 1.1-2, URL https://CRAN.R-project.org/package=RColorBrewer.

Nychka D, Furrer R, Paige J, Sain S (2018). ***fields**: Tools for Spatial Data*. doi:10.5065/d6w957ct. R package version 9.6, URL https://CRAN.R-project.org/package=fields.

R Core Team (2018). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.

Rowlingson B, Diggle P (2017). ***splancs**: Spatial and Space-Time Point Pattern Analysis*. R package version 2.01-40, URL https://CRAN.R-project.org/package=splancs.

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

Sain SR, Furrer R, Cressie N (2011). "A Spatial Analysis of Multivariate Output from Regional Climate Models." *The Annals of Applied Statistics*, **5**, 150–175. doi:10.1214/10-aoas369.

Wallin J, Bolin D (2015). "Geostatistical Modelling Using Non-Gaussian Matérn Fields." *Scandinavian Journal of Statistics*, **42**, 872–890. doi:10.1111/sjos.12141.

Wiesenfarth M, Krivobokova T, Klasen S, Sperlich S (2012). "Direct Simultaneous Inference in Additive Models and Its Application to Model Undernutrition." *Journal of the American Statistical Association*, **107**(500), 1286–1296. doi:10.1080/01621459.2012.682809.

**Affiliation:**

David Bolin
Department of Mathematical Sciences
Chalmers University of Technology
S-412 96 Gothenburg, Sweden
E-mail: david.bolin@chalmers.se
URL: http://www.math.chalmers.se/~bodavid/

Finn Lindgren
The University of Edinburgh
James Clerk Maxwell Building
Peter Guthrie Tait Road
Edinburgh EH9 3FD, United Kingdom
E-mail: finn.lindgren@ed.ac.uk
URL: http://www.maths.ed.ac.uk/~flindgre/

*Journal of Statistical Software* http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
August 2018, Volume 86, Issue 5 *Submitted:* 2016-12-13
doi:10.18637/jss.v086.i05 *Accepted:* 2017-08-14