inlabru: an R package for Bayesian spatial modelling from ecological survey data

Citation for published version:
Bachl, F, Lindgren, F, Borchers, DL & Illian, JB 2019, 'inlabru: an R package for Bayesian spatial modelling from ecological survey data', *Methods in ecology and evolution*. https://doi.org/10.1111/2041-210X.13168

Digital Object Identifier (DOI):
10.1111/2041-210X.13168

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
Methods in ecology and evolution

inlabru: an R package for Bayesian spatial modelling from ecological survey data

Fabian E. Bachl[1], Finn Lindgren[1], David L. Borchers[2,*], and Janine B. Illian[2]

[1]School of Mathematics, James Clerk Maxwell Building, The King's Buildings, Peter Guthrie Tait Road City, Edinburgh, EH9 3FD, Scotland

[2]Centre for Research into Ecological and Environmental Modelling, School of Mathematics and Statistics, University of St Andrews, The Observatory, St Andrews, Fife, KY16 9LZ, Scotland

[*]Corresponding author: dlb@st-andrews.ac.uk

1 Summary

1. Spatial processes are central to many ecological processes, but fitting models that incorporate spatial correlation to data from ecological surveys is computationally challenging. This is particularly true of point pattern data (in which the primary data are the locations at which target species are found), but also true of gridded data, and of georeferenced samples from continuous spatial fields.

2. We describe here the `R` package `inlabru` that builds on the widely-used `R-INLA` package to provide easier access to Bayesian inference from spatial point process, spatial count, gridded, and georeferenced data, using integrated nested Laplace approximation (INLA, Rue *et al.*, 2009).

3. The package povides methods for fitting spatial density surfaces and estimating abundance, as well as for plotting and prediction. It accommodates data that are points, counts, georeferenced samples, or distance sampling data.

4. This paper describes the main features of the package, illustrated by fitting models to the gorilla nest data contained in the package `spatstat` (Baddeley & Turner, 2005), a line transect survey data set contained in the package `dsm` (Miller *et al.*, 2018), and to georeferenced sample from a simulated continuous spatial field.

Keywords: Spatial modeling, point process, spatial count, georeferenced data, Bayesian inference

2 Introduction

Many ecological datasets exhibit spatial correlation in observed variables, due to biotic or abiotic processes such as dispersal limitation, social aggregation, and spatial structure in unobserved explanatory variables. Whether the observations are points (e.g. animal locations), counts (e.g. the numbers of animals in spatial samples) or values of some continuous variable (e.g. nutrient levels at sampled points), spatial correlation causes every observation to depend on every other observation within some unknown correlation range. Dealing with this requires models that are mathematically more complex and computationally more demanding than is the case when there is independence among observations.

We account for spatial dependence by incorporating a Gaussian random field (GRF) into models. GRFs are spatially continuous random processes in which random variables at any point in space are normally distributed and are correlated with random variables at other points in space according to a continuous correlation process. GRFs provide a means of modelling the spatial signal in the observations that cannot be accounted for by covariates.

In the case of point data and count data, the GRF is linked to the response variable by a log link function, to give a log Gaussian Cox process (LGCP) model (Møller & Waagepetersen, 2007). (Called "log Gaussian" because the log of the intensity at any point is assumed to be normally distributed, and "Cox process" because this is a Poisson process that has a randomly varying intensity function.) What spatial statisticians call the "intensity" is the density in our context, and we will use the term "density" for this henceforth.

The GRF is approximated by the solution to a stochastic partial differential equation (SPDE; see Lindgren *et al.*, 2011, for details). We do not have space to describe the details of SPDEs, but fortunately the mathematical details need not be understood to use them in `inlabru`. It is sufficient to know that SPDEs provide an efficient way of approximating the GRF in continuous space (Simpson *et al.*, 2016).

Integrated nested Laplace approximation (INLA) Bayesian methods (Rue *et al.*, 2009) are used for inference. INLA is a fast and accurate alternative to Markov chain Monte Carlo (MCMC) for fitting latent Gaussian models, i.e., hierarchical models in which there are unobserved (latent) normally distributed random variables. The models we consider here, in which the GRF is latent, are of this type. We refer the reader to the "Gentle INLA tutorial" at `https://www.precision-analytics.ca/blog-1/inla` for more about INLA, and to the R-INLA project at `http://www.r-inla.org/` for more about the `R-INLA` package on which the `inlabru` package builds.

The `R-INLA` package currently requires users to have knowledge of likelihood approximation schemes, and does not allow inference when detection probability is unknown, as is common in many wildlife surveys. The `inlabru` package makes fitting spatial models with INLA more accessible to non-specialist users by employing simpler syntax, and it extends the class of models that can be fitted to include distance sampling.

We illustrate the scope of the package by fitting models to point and count data from a survey of gorilla (*Gorilla gorilla*) nests by Funwi-Gabga (2008), a line transect survey of pantropical spotted dolphins (*Stenella*

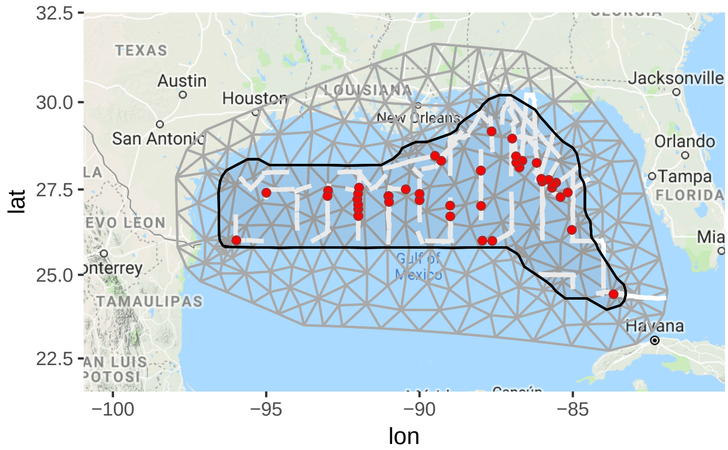

Figure 1: Pantropical dolphin survey data plotted using `ggmap` and the `gm` method. Grey triangles show the `inla.mesh` object. The survey region boundary (black) is held in a `SpatialPolygonsDataFrame`. The line transects (white lines) are held in a `SpatialLinesDataFrame` and the detected dolphins (red points) are held in a `SpatialPointsDataFrame`.

attenuata) [1], and a simulated survey of a continuous spatial field. Other examples can be found at `http://inlabru.org/tutorials`.

3 Data format and visualization

The `inlabru` package supports the `sp` package data structures (Pebesma & Bivand, 2005). These are well documented within `sp`, togeher with powerful functions for manipulating them. The `SpatialPointsDataFrame` structure stores spatial points together with spatial covariate data and attributes of points (e.g. size or species). `SpatialLinesDataFrame`s store spatial data for line transect surveys and `SpatialPolygonsDataFrame`s are used to define survey regions and sample plots.

Continuous space is approximated in `inlabru` using a "mesh" (a tiling of space with triangular tiles – see Figure 1 for example). We use the `inla.mesh` class of object from the `INLA` package for this approximation.

Data visualization tools in `inlabru` are built on the `ggplot2` (Wickham, 2009) and `ggmap` (Kahle & Wickham, 2013) packages, with customized `inlabru` functions such as `gg` and `gm` to extend their functionality. Figure 1 shows an example of such a plot generated from a line transect survey of pantropical spotted dolphins in the Gulf of Mexico.

4 Key syntax

Models are defined by specifying

1. a `formula` for the linear or nonlinear predictor that defines the log density function,
2. the components of this predictor (one of which is typically an SPDE), and
3. the observed variable distribution.

[1] see `http://seamap.env.duke.edu/dataset/25`) for details of this survey

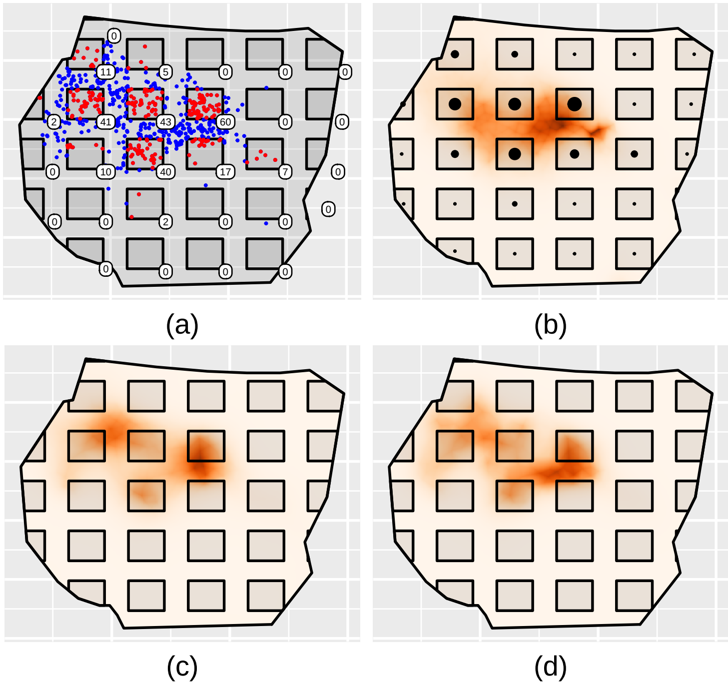

Figure 2: Analysis of gorilla nests as a count and as a point process model. Panel (a) depicts the survey region, search plots, undetected (blue) and detected (red) nests, including the nest counts (white boxes). Panel (b) shows a density fit with `bru` to nest counts, associating counts with the plot centres. Panels (c) and (d) show point process fits obtained with `lgcp` using nests within the plots, and all, respectively.

Models are fitted using the function `bru( )` or, for LGCP models, `lgcp( )`. Examples are given below.

5 Spatial count data

We begin by using `inlabru` to infer a smooth spatial density surface from plot samples in which the response is the count of gorilla nests in each plot (see Figure 2 (a)). Although the exact locations of all nests were recorded, we initially use only nest counts in a sample of plots. The R code showing how to load the package and the data is provided as Supporting Information S1.

The observed response, `count`, is the number of nests in a plot, which we assume to be a Poisson random variable. We also assume that the log density of the Poisson distribution varies in space and is the sum of an intercept term (the base log density) and an SPDE (which captures the spatially correlated variation about the base). We name the SPDE `spat`[2]. Recall that the SPDE approximates a GRF, and we specify below that the correlation of this field has a Matérn correlation structure. This correlation model (with unknown parameters) is specified using the `INLA` function `inla.spde2.matern`. The SPDE and correlation model are defined on a `mesh`, which we do not show here because it obscures important elements of the plots (see Figure 1 for an example of a mesh).

The two components of our linear predictor are the intercept and the SPDE. We store these in an object called `cmp` as follows:

[2]This name can be chosen by the user.

```
cmp <- count ~ spat(map = coordinates, model = inla.spde2.matern(mesh)) + Intercept
```

The syntax for defining SPDEs requires a name for the SPDE ("spat" here), followed by specification, in brackets, of the domain on which it is defined ("map=coordinates" here), and its correlation function ("model=inla.spde2.matern(mesh)" here). Note that coordinates is a method defined by the package sp to extract locations from sp spatial objects. Using it as above specifies that the SPDE applies to spatial coordinates.

We use the inlabru function bru to fit the model to the gorilla count data gcounts (a SpatialPointsDataFrame with a data field count containing the nest count data):

```
fit <- bru(components = cmp,
           family = "poisson",
           data = gcounts,
           formula = ~ spat + Intercept,
           options = list(E = gcounts$exposure))
```

The components parameter specifies the model components. The family parameter specifies the probability density function (PDF) of the response. (All family types supported by the INLA package are supported by inlabru.) The formula specifies how the components are combined to create a linear (in this case) predictor for density. The parameter E in the options list sets the "exposure" parameter of the Poisson family, namely the areas of each searched plot in this example. (The log of the exposure would be an offset in a Poisson generalised linear model.)

We did not need to specify the formula above, because inlabru assumes that it is the sum of the components if no formula is given. The formula is really only required when it is not this sum (see examples in Sections 6.2 and 6.3 below).

We can predict any function of any subset of the components of the model specification (cmp above) using inlabru's predict function. For example, predictions of the density are obtained as follows:

```
pxl <- pixels(mesh, mask = boundary)
dens <- predict(fit, pxl, formula = ~ exp(spat + Intercept))
```

The first line creates a regular grid of locations covering the survey region. The third argument of the predict call specifies what is to be predicted, as a function of the components. To predict on the scale of the linear predictor, for example, we would just replace exp(spat+Intercept) with spat+Intercept. The predict function estimates the posterior densities of whatever function is specified in its formula argument.

The object obtained from predict is a SpatialPixelsDataFrame. As with any other spatial object, we can employ the gg function to add it to a blank plot. Hence, calling ggplot() + gg(dens) will render the density shown in Figure 2 (b).

6 Fitting point processes

We now consider the case in which the data are the *locations* of nests within plots. Some information about the spatial process governing nest locations is lost when locations are aggregated into counts within plots, and we would like to use all the information in the data. In this case, the response variables are the coordinates of the individual nests, and the locations are random variables, whereas with count data the locations of the plots were fixed and known and the counts were random variables. Spatial point processes models (Møller & Waagepetersen, 2007; Illian *et al.*, 2008; Baddeley *et al.*, 2015) are used when the points themselves are the random variables. More specifically, we use an LGCP, in which the log density includes a GRF, to model overdispersion and clustering that cannot be accounted for by covariates.

6.1 Inference for spatial Poisson point processes

The work flow of inference in point processes fitting is similar to that described above. We specify the model by replacing the user-defined response "`count`" on the left of the component specification, with the key word "`coordinates`" to indicate that the responses are spatial coordinates.

```
cmp <- coordinates ~ spat(map = coordinates, model = inla.spde2.matern(mesh)) + Intercept
```

The R code showing how to load the data is provided in Supporting Information S1. Fitting an LGCP model is done using `lgcp`:

```
fit <- lgcp(components = cmp, data = plotnests, samplers = plots)
```

Here `plotnests` is a `SpatialPointsDataFrame` containing the locations of the observed nests. The `samplers` argument is passed a `SpatialPolygonsDataFrame` called `plots` that specifies the polygons that were searched. If this argument is left empty, `lgcp` will assume that the whole domain defined by the mesh (contained in the SPDE specification, `spat`, in `cmp`) was searched, which would result in biased inference if the whole domain was not searched.

Running the code above and then using `predict` and `plot` yields the density plot shown in Figure 2 (c). For comparison, Figure 2 (d) shows a LGCP fit to the complete gorilla nest data set, which was obtained as above but with `samplers=boundary` in place of `samplers=plots`, where `boundary` is a `SpatialPolygonsDataFrame` object defining the survey boundary.

6.2 Inference for univariate point processes: distance sampling detection function

We illustrate `inlabru`'s ability to model one-dimensional point processes by fitting a detection function to the perpendicular distances of detected dolphins on the line transect survey shown in Figure 1. The `R` code showing how to load and prepare the data is provided as Supporting Information S2.

The observed density of distances to detections is the product of the underlying density of distances to dolphins ($\lambda(d)$ say, where d is distance) and the probability of detecting a dolphin that is at distance d ($h(d; \log\{\sigma\})$

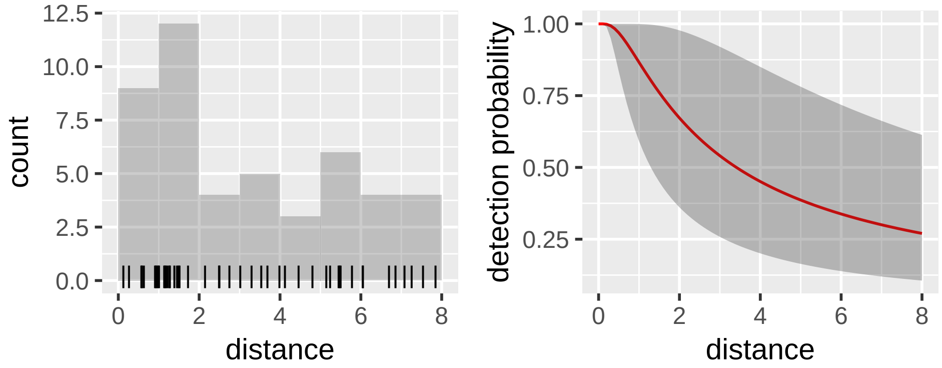

Figure 3: Pantropical dolphin detection distances (left) and fitted hazard rate detection function (right), showing 95% credibile region. With adequate fit, the red line is a smooth through the histogram, as is apparent here.

say, where $\log\{\sigma\}$ is an unknown parameter). Under the usual line transect assumption that animals are uniformly distributed with respect to distance from the line, $\lambda(d) = \lambda$ so that the density of the observed distance process is $h(d; \log\{\sigma\})\lambda$. Hence the log density can be written as $\log[h(d; \log\{\sigma\})] + \beta_0$, where $\lambda = e^{\beta_0}$.

We specify the (nonlinear) predictor for this model, and its components, as follows:

```
fml <- distance ~ log(h(distance, lsig)) + Intercept
cmp <- distance ~ lsig + Intercept
```

where `h(distance,lsig)` is $h(d, \log\{\sigma\})$ and `Intercept` is $\beta_0 = \log(\lambda)$. To complete the specification we need to define the function `h(distance,lsig)`. We define it to be the hazard-rate detection function of Hayes & Buckland (1983), with shape parameter 1, as follows:

```
h <- function(distance, lsig){ 1-exp(-(distance/(exp(lsig)))^-1)}.
```

Because one of the components (the parameter `lsig`) enters the linear predictor for log density via a non-linear function, $\log[h(d; \log\{\sigma\})]$, we need to specify the `formula` explicitly, rather than have `inlabru` construct it by default as the sum of the `components`. This model is fitted using `lgcp` as follows:

```
fit <- lgcp(cmp, mexdolphin$points, formula = fml).
```

where `mexdolphin$points` is a `SpatialPointsDataFrame` with a variable `distance` for every point.

After fitting the model, predicting the detection function for distances 0 to 8 (the maximum distance considered) is straightforward using

```
pts <- data.frame(distance = seq(0,8, by = 0.1)),
dfun <- predict(fit, pts, formula = ~ h(distance, lsig)
```

while `plot(dfun)` plots it with 95% credible interval (as shown in Figure 3).

We note in passing that `inlabru` can be used to estimate *any* PDF using commands similar to those above, if we consider the intensity of a Poisson process to be an unnormalized PDF.

6.3 Inference for thinned Poisson processes: distance sampling

We now use `inlabru` to estimate the density and distribution of dolphin groups with the conventional distance sampling assumption of uniform group distribution within searched strips. This assumption is tenable because the searched strips have negligible width compared to the size of the survey region (see Figure 1) and were laid down with random start location. We implement the assumption by simultaneously modelling the spatial distribution of detected points (as in Section 6.1) and the PDF of distances of detections from the lines, assuming uniform distribution of these distances (as in Section 6.2). The R code for this is provided as Supporting Information S4.

An analysis of these data (also assuming uniform group distribution within searched strips) using the R package `dsm` is available at `http://distancesampling.org/R/vignettes/mexico-analysis.html`. The methods implemented in `inlabru` and `dsm` differ in a number of ways, including that `inlabru` implements a fully-Bayesian approach, so one can specify priors on parameters (not illustrated here), and `inlabru` estimates detection probability and the density surface simultaneously, while `dsm` estimates detection probability in one step and the density surface conditional on this estimate, in another.

The key to simultaneous estimation of detection probability and the density surface is the fact that if the locations of points arise from a Poisson process, then the locations of the *detected* points arise from a *thinned* Poisson process. "Thinning" involves detecting points with some probability (h, say) that is less than 1. The density (intensity) of a thinned Poisson process is the unthinned density D, multiplied by the thinning probability h. For example, if $h = 0.5$ so that half the points are detected on average, then the density of detected points is half that of the all points: $Dh = D/2$. On a line transect survey, the probability of missing å point depends on its distance d from the line, so that h is a function of distance ($h(d)$) and the density of the thinned Poisson process at the point's location is $Dh(d)$, where D is the underlying density at this location. Writing D as $D = \exp(\text{Intercept})$ and noting that $Dh(d) = \exp(\text{Intercept}+\log(h(d))$, we see that the log density of the thinned Poisson process is equal to the log density of the underlying process plus the log of the detection probability. This is convenient, because it means that we can do inference for thinned LGCPs by simply adding a term for the thinning probability to the log density.

With this in mind, and noting that the thinning probability has an unknown parameter that we call `lsig`, we specify our model by combining the `components` specification and `formula` specifications from Sections 6.1 and 6.2.

```
cmp <- ~ spat(map = coordinates, model = inla.spde2.matern(mesh)) + lsig + Intercept
fml <- coordinates + distance ~ spat + log(h(distance, lsig)) + log(1/8) + Intercept
```

The left hand side of the formula (`coordinates + distance`) tells `inlabru` that we are modelling both the spatial point process governing dolphin group locations, and the detection distances. The right hand side says that the log density of this process is the sum of the log detectability and the spatial process composed of the spatial SPDE, and the `Intercept`. The offset term $\log(1/8)$ specifies that the density of distances is assumed

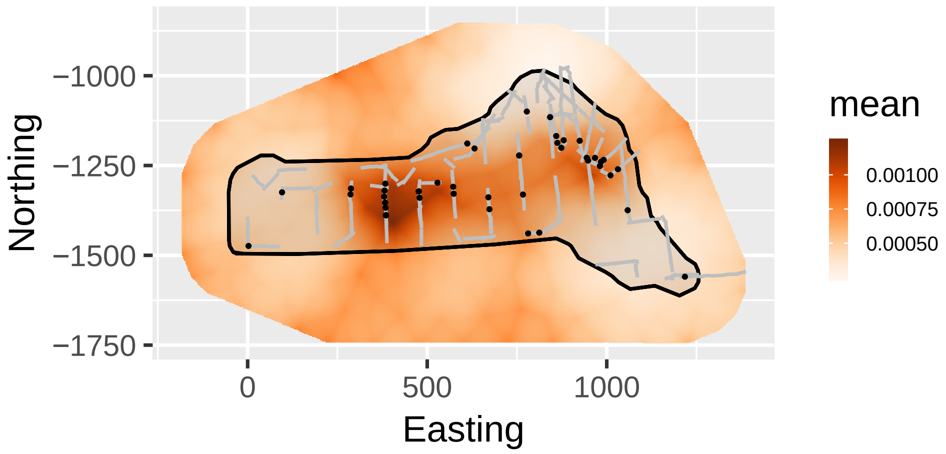

Figure 4: Predicted density surface (in counts per square km) of a point process model for dolphin groups fitted jointly with a hazard rate detection function (not shown).

to be constant on the distance interval $(0, 8)$ – as the transect half-width is 8 km.

With the above definitions, fitting the model is straightforward using the same syntax as shown in Section 6.1, where now the `samplers` argument is a `SpatialLinesDataFrame` storing the survey's ship transects. The prediction code introduced in Section 5 is then used to estimate the spatial density surface shown in Figure 4(a).

We can add further processes, such as a group size probability model. This allows us to make detection probability depend on group size and to model a spatially varying group size distribution. We do not illustrate this here for lack of space.

7 Georeferenced data from a continuous spatial field

We illustrate spatial modelling from a continuous spatial field by sampling the simulated field (which might correspond to a soil nutrient level, for example) shown in Figure 5(a), at the locations of the crosses in that figure. Having specified a Matérn correlation function using `inla.spde2.matern` in a similar way to that shown previously, and given that the sampled observations are in the `observed` data field of a `SpatialPointsDataFrame` named `geosamp`, the model is fitted as follows, assuming a Gaussian error model:

```
cmp <- observed ~ field(map = coordinates, model = inla.spde2.pcmatern(mesh)) + Intercept
fit <- bru(components = cmp, data = geosamp, family = "gaussian")
```

(Here we have named the SPDE "`field`" rather than "`spat`".)

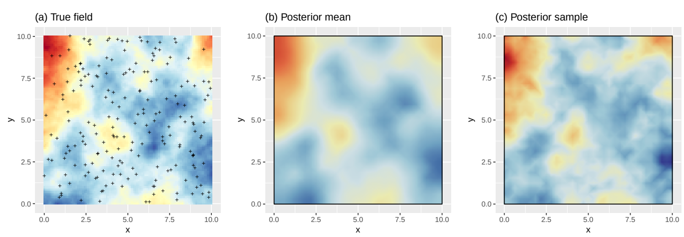

Figure 5: (a) A simulated continuous spatial field, showing sample locations, (b) the posterior mean of the model fitted to the sample data, and (c) a sample from the posterior distribution of the field.

The mean of the fitted model is shown in Figure 5(b), while a sample from the posterior distribution of the field is shown in Figure 5(c). Note that the mean surface is necessarily smoother than the true field (which is conceptually a draw from a random field with the given mean), while the posterior sample better reflects the fine-scale structure of the true field.

8 Discussion

The inlabru package makes Bayesian spatial modelling with INLA, including point process modelling, more accessible to ecologists. It allows one to model species distribution and estimate density and abundance with data that are (a) complete spatial maps of the locations of individuals or groups, (b) counts in plots, (c) points, and (d) distance sampling data.

It is distinguished from methods and software that fit density surfaces to count data in that it can deal with points as responses in continuous space and does not require that space be discretised (although inlabru can deal with such data, as illustrated in Section 5 above). Nor does it require a neighbourhood structure to be defined, as is required for conditional autoregressive models or simultaneous autoregressive models, for example.

It also provides a means of doing Bayesian spatial modelling with distance sampling data. Its distance sampling capabilities are not as well developed as those of the frequentist package dsm (Miller *et al.*, 2018), and unlike dsm, it estimates the detection probability and density surface simultaneously. It shares this feature with the frequentist package unmarked (Fiske & Chandler, 2011), although unmarked has no spatial modelling capabilities. Simultaneous estimation of detection probability and the density surface is conceptually satisfying, but the jury is out on whether this, or estimation of the two in separate steps, is preferable in practice.

Features of inlabru that we do not have space to describe include its ability to do temporal and spatio-temporal modelling and its ability to simultaneously estimate the density of a point process and the spatially-varying density of what spatial statisticians call “marks” on points (dolphin group size, being an example) as well as its impact on the shape of the detection function.

Features under development include point transect data, modelling multi-species density when there is spatial interaction or common explanatory environmental data for the distribution of different species sharing a habitat, and modelling of habitat preference based on telemetry data. There are some technical obstacles to implementing spatial capture-recapture methods (Efford, 2004; Borchers & Efford, 2008; Royle & Young, 2008) in inlabru, but work in this area is ongoing.

Acknowledgments

This research was funded by EPSRC grants EP/K041061/1, EP/K041053/1, and EP/K041053/2.

Supplementary Material

RMarkdown scripts:

- Supporting Information S1: `1_spatial_gorilla_models.Rmd`. Code for spatial Poisson count and LGCP inference.
- Supporting Information S2: `2_dfun_univariate.Rmd`. Code for detection function inference.
- Supporting Information S3: `3_distsamp.Rmd`. Code for line transect models.
- Supporting Information S4: `4_georefsim.Rmd` Code for models for georeferenced data.

Author contributions statement

FB, DB, JI and FL conceived the ideas and designed methodology; FB and DB analysed the data; FB and FL wrote the code, with a minor contribution from DB; FB led the writing of the manuscript, with major contributions from all authors.

References

Baddeley, A., Rubak, E. & Turner, R. (2015) *Spatial point patterns: methodology and applications with R.* CRC Press.

Baddeley, A. & Turner, R. (2005) spatstat: An R package for analyzing spatial point patterns. *Journal of Statistical Software*, **12**, 1–42.

Borchers, D.L. & Efford, M.G. (2008) Spatially explicit maximum likelihood methods for capture-recapture studies. *Biometrics*, **64**, 377–385.

Efford, M.G. (2004) Density estimation in live-trapping studies. *Oikos*, **106**, 598–610.

Fiske, I. & Chandler, R. (2011) unmarked: An R package for fitting hierarchical models of wildlife occurrence and abundance. *Journal of Statistical Software*, **43**, 1–23.

Funwi-Gabga, N. (2008) *A pastoralist survey and fire impact assessment in the Kagwene Gorilla Sanctuary.* Master's thesis, Geology and Environmental Science, University of Buea, Cameroon.

Hayes, R.J. & Buckland, S.T. (1983) Radial-distance models for the line-transect method. *Biometrics*, **39(1)**, 29–42.

Illian, J.B., Penttinen, A., Stoyan, H. & Stoyan, D. (2008) *Statistical Analysis and Modelling of Spatial Point Patterns.* Wiley, Chichester.

Kahle, D. & Wickham, H. (2013) ggmap: Spatial visualization with ggplot2. *The R Journal*, **5**, 144–161.

Lindgren, F., Rue, H. & Lindström, J. (2011) An explicit link between Gaussian fields and Gaussian Markov random fields: the SPDE approach (with discussion). *Journal of the Royal Statistical Society: Series B*, **73(4)**, 423–498.

Miller, D.L., Rexstad, E., Burt, L., Bravington, M.V. & Hedley., S. (2018) *dsm: Density Surface Modelling of Distance Sampling Data.* R package version 2.2.16.

Møller, J. & Waagepetersen, R.P. (2007) Modern statistics for spatial point processes (with discussion). *Scandinavian Journal of Statistics*, **34**, 643–711.

Pebesma, E.J. & Bivand, R.S. (2005) Classes and methods for spatial data in R. *R News*, **5**, 9–13.

Royle, J.A. & Young, K.V. (2008) A hierarchical model for spatial capture-recapture data. *Ecology*, **89**, 2281–2289.

Rue, H., Martino, S. & Chopin, N. (2009) Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations (with discussion). *Journal of the Royal Statistical Society: Series B*, **71(2)**, 319–392.

Simpson, D., Illian, J.B., Lindgren, F., Sørbye, S.H. & Rue, H. (2016) Going off grid: Computationally efficient inference for log-Gaussian Cox processes. *Biometrika*, **103**, 49–70.

Wickham, H. (2009) *ggplot2: Elegant Graphics for Data Analysis.* Springer-Verlag New York.