




The ppmData R-package for setting up spatial point process models

Skipton N.C. Woolley^{1, 2} and Scott D. Foster³

¹ School of Ecosystems and Forestry Science, The University of Melbourne ² Oceans & Atmosphere, CSIRO ³ Data61, CSIRO

DOI:

Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Submitted:

Published:

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC-BY](#)).

Summary

Spatial data, the locations of objects in a spatial region, is a common type of data recorded in ecology, environmental sciences and epidemiology amongst many others. An appropriate statistical approach for such data are spatial point processes models (Baddeley & Turner, 2000; e.g. Cressie, 1993). Spatial point processes can be used to understand the location of objects to inform decision making and improve ecological, health and other socio-economic outcomes. For example, the locations of individuals from a species' population are typically not distributed completely at random, but rather are observed at locations with favorable habitat conditions for that species. We might expect these species sightings are distributed according to relevant covariates.

Inhomogeneous Poisson Point Process models (IPPM) allow for the spatial intensity of a point process to vary across space, usually as a function of spatially reference covariates. One challenge when fitting an inhomogeneous Poisson point process is the integral used to describe the intensity surface in the log-likelihood (as described below) has no known closed analytic solution. For IPPMs, the Berman-Turner device (Berman & Turner, 1992) is commonly used to approximate the model using quadrature within a weighted Poisson generalized linear model. The type of quadrature scheme can have important ramifications for overall model fitting and inference (Warton & Shepherd, 2010). Approximating the integral accurately often comes at an increased computation cost, where a large number of quadrature points are required to robustly estimate the integral (Renner et al., 2015).

Our `ppmData` package was setup to use a quasi-random quadrature approach as an alternative to pseudo-random sampling (Phillips et al., 2009) and regular grid (Warton & Shepherd, 2010) quadrature approaches. Quasi-random sampling, a base-case of Balanced Acceptance Sampling (Robertson, Brown, McDonald, & Jaksons, 2013), can be used to efficiently perform numerical integration (Halton, 1960) and create spatially balanced survey designs. Quasi-random sampling is an efficient form of spatial sampling as it approximately balances over all spatially smooth covariates (Grafström & Tillé, 2013), even if they are not included or considered in the quadrature generation. When used in the fitting of an inhomogeneous poisson point process, quasi-random (spatially balanced) quadrature also reduces the influence of spatial autocorrelation between quadrature points and is likely to improve numerical approximation of the intensity (Foster et al., 2017; Liu & Vanhatalo, 2020).

Here we present the `ppmData` R package that is designed to setup a quadrature scheme using Dirichlet tessellation for fitting spatial point process models. Our initial intention was to develop multiple species (marked) quadrature in order to improve computation of multiple species Poisson point process models. Since then, the package has involved into one which can be useful for fitting univariate response inhomogeneous spatial point process, as discussed in this paper. We describe a how to set up a inhomogeneous Poisson point process quadrature and provide a simple demonstration on how one can fit an

inhomogeneous Poisson point process using a `ppmData` object.

Method

Inhomogeneous Poisson Point Processes (IPPM)

An IPPM describes the pattern in which n observed locations, $(\mathbf{y} = (y(\mathbf{s}_1), y(\mathbf{s}_2), \dots, y(\mathbf{s}_n)))$ say, makes over a spatial window $\mathcal{A} \subset R^2$. An IPPM is based on the assumption that the number of points, n , is a Poisson random variable, with spatially varying intensity $\Lambda = \int_{\mathbf{s} \in \mathcal{A}} \lambda(\mathbf{s}) d\mathbf{s}$, and that the spatial locations of \mathbf{y} are independently drawn from the spatial surface (e.g. Cressie, 1993). For convenience, we define (for example) $\lambda(\mathbf{s}_i)$ as the intensity surface of the point process at the i^{th} location \mathbf{s}_i . The intensity surface can be defined as $\log\{\lambda_i\} = \mathbf{x}_i^T \boldsymbol{\beta}$ where \mathbf{x}_i is a vector of spatially located covariates and $\boldsymbol{\beta}$ is a corresponding vector of coefficients.

The approximate log-likelihood $\ell(\boldsymbol{\beta}; \{\mathbf{x}_i\})$ can be written as a weighted Poisson likelihood (Berman & Turner, 1992)

$$\begin{aligned} \ell(\boldsymbol{\beta}|\mathbf{y}) &= \sum_{i=1}^n \log(\lambda_i) - \int_{\mathbf{s} \in \mathcal{A}} \lambda(\mathbf{s}) d\mathbf{s} - \log(n!) \\ &\approx \sum_{i=1}^n \log(\lambda_i) - \sum_{i=1}^m w_i \lambda_i \\ &= \sum_{i=1}^m w_i \{z_i \log(\lambda_i) - \lambda_i\} \end{aligned} \quad (1)$$

where z_i is an indicator variable identifying if site i is one of the n observed presences or one of the m quadrature point, w_i stores the quadrature weights, which sum to total area of the region $|\mathcal{A}|$. In (1), we can see the integral in the log-likelihood $\ell(\boldsymbol{\beta}|\mathbf{y})$ is being estimated using quadrature (Baddeley & Turner, 2000; Diggle, Menezes, & Su, 2010). Warton & Shepherd (2010) demonstrated for presence-only species distribution models, that using numerical quadrature with a point process framework makes the models scale invariant and treats the quadrature purely as a tool for approximating the numerical integral.

Established methods to approximate the log-likelihood using quadrature

There are a number of ways to setup the quadrature scheme to get an approximation of the integral $\Lambda = \int_{\mathbf{s} \in \mathcal{A}} \lambda(\mathbf{s}) d\mathbf{s}$. One common, and arguably the easiest to program is to use a regular grid (Warton & Shepherd, 2010). This approach essentially creates a regular grid at a known resolution within a spatial window ($\mathcal{A} \subset R^2$). The grid approach is typically dense; and as we double the resolution it results in 4-times the number of quadrature points. The grid approach is not without detraction though. In particular, it is unlikely to handle edge effects well (when there is a consistent and potentially large pattern near the boundaries of \mathcal{A}). Then the grid will either be defined away from the edge, missing important trends, or defined on the edge, over-representing this trend. Also, another potential problem stems from the unlikely event that the spatial surface $\lambda(\mathbf{s}_i)$ has periodicity in its pattern aligned to the direction of the grid, such as repeated locations of mountain ridges. In that case, the grid-based quadrature will over- or under-estimate the integral depending on whether the grid is coincides with peaks or troughs of the surface.

Another common approach is to use a pseudo-random spatial samples and generate approximate areal weights per quadrature point (Phillips et al., 2009; Renner et al., 2015).

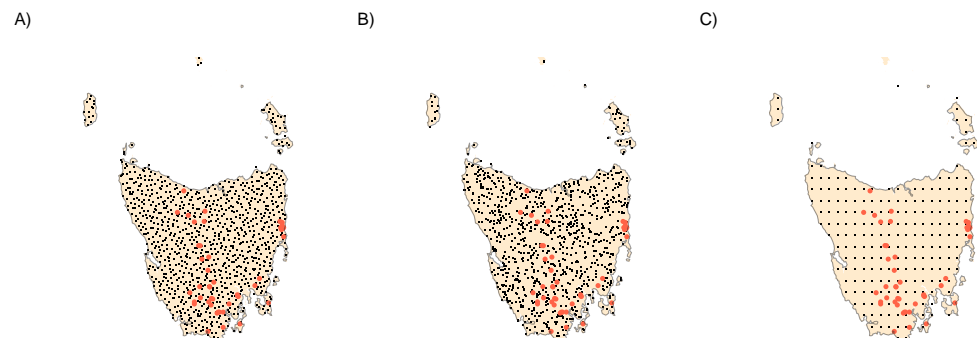


Figure 1: Quadrature schemes generated using the ppmData package for the species *Tasmaphena sinclairi* using Tasmania, Australia, as our spatial region. Red points are the known presences of *Tasmaphena sinclairi*. Black points are the background quadrature locations. A) Quasi-random quadrature where the integration points are generated using a quasi-random areal sample. Weights for each integration point are calculated as the dual of the subsequent Dirichlet tessellation of both the presence and integration points. B) Pseudo-random quadrature generated using random points from within the window. C) Grid quadrature where quadrature points are generated on a regular grid.

Typically, this is done by generating pseudo-random points within the study window $\mathcal{A} \subset \mathbb{R}^2$ and assuming that all quadrature points have equal weights, generally calculated as $w_i = \frac{m}{|\mathcal{A}|}$ (Renner et al., 2015). This approach is unlikely to suffer either of the potential problems from the grid approach, however it may be inefficient – requiring more points to achieve the same level of accuracy (e.g. Robertson et al., 2013).

Here we use the quasi-random quadrature as a method to trade-off the robustness of pseudo-random sampling and the efficiency of the grid. Quasi-random quadrature spreads the quadrature points in space but simultaneously retains some of the important properties of random sampling. Quasi-random quadrature schemes are available in other R packages, like the excellent `spatstat` (Baddeley, Rubak, & Turner, 2015). The `rQuasi` function which can be used for generating a quasi-random spatial sample within a window. However, if the window is non a rectangle, the number of points will be reduced to include only the ones remaining inside the window boundary, so specifying the number of active points is guess work. Our package ppmData allows the exact number of quadrature points to be specified within a spatial window.

The package `spatstat` also allows a user to develop a quasi-random quadrature scheme, but this appears to be quite slow when used with Dirichlet weights for larger numbers of dummy (quadrature) points (> 10000). For example, a regular window with 100000 quadrature points takes approximately 3.5 minutes to run in `spatstat` using `quadscheme` with `method=Dirichlet`. While our package computes the same Dirichlet tessellation in about 4.5 seconds. The reason for this is due to the computational complexities of calculating Dirichlet weights for large numbers of quadrature points. The computation increases super-linearly with the number of points. Part of the challenge when developing ppmData was to make a code which made this step efficient. We approached this by producing C++ code to undertake a Delaunay Triangulation using the radial sweep algorithm (Sinclair, 2016). The C++ code is run on each subset of \mathcal{A} for presences and quadrature sites. We then calculate the Dirichlet tessellations as the dual graph of this triangulation. This makes Dirichlet tessellation more efficient and faster for larger quadrature schemes. The Dirichlet weights of each point within the quadrature (presences &

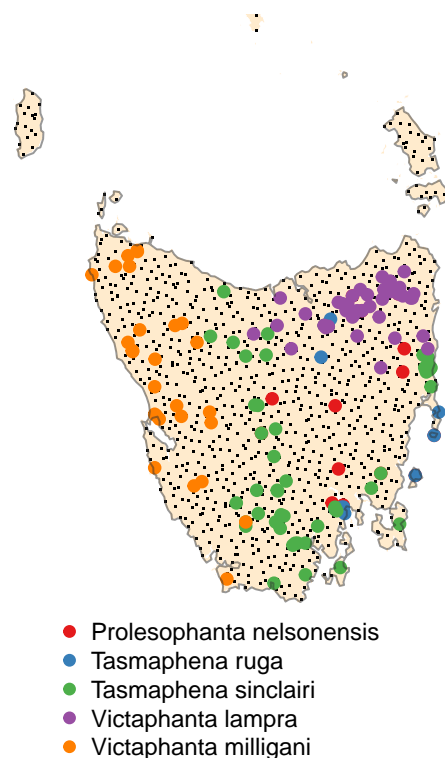


Figure 2: Multiple species quadrature scheme. Here we can see the presences of five different snail species across Tasmania. For the multiple species quadrature there is a common set of background quadrature sites, but the object returns species specific weights.

background points) are calculated as the area of each respective Dirichlet tessellation.

Generate a quadrature scheme using quasi-random sampling

In the `ppmData` package we have tried to make the generation of a quadrature scheme of a single or marked point process as easy as possible. We provide a simple interface for generating a Poisson point process data object and base all inputs and outputs using `terra` (Hijmans, 2021) and `sf` (Pebesma, 2018) packages. All we need to pass is a set of coordinates from a point process and we can generate a very basic point process data object. Ideally, the user knows the region they are interested in modelling and this is passed as a window, we use a `terra` `SpatRaster` object. The window defines the region where we expect to encounter observations and where to generate the quadrature locations. There are a number of other calls which can be passed to `ppmData`, but the main one is the inclusion of covariates which can be used in an inhomogeneous Poisson point process. Under this scenario pass a set of covariates that share the same extent and resolution as the window `SpatRaster`. The values of the covariates will be extracted at the locations of the quadrature scheme. Whilst not distinguishing for this package, the `ppmData` package can also generate grid and pseudo-random quadrature schemes as presented in Fig. 1.

Multiple species (marked) point process quadrature

One of the main reasons for creating the `ppmData` package was to develop quadrature schemes for multiple species (marked) point process. The multiple species quadrature scheme can be used in multiple species models available for use in the `ecomix` package (Woolley, Foster, & Dunstan, 2022), or models like joint species distribution models (Ovaskainen & Soininen, 2011). Here is a quick example of how we would set up a quasi-random quadrature scheme for multiple species. The main difference is the the column `speciesID` contains multiple species, `ppmData` recognises this internally and subsequently sets up a marked point process quadrature scheme. The quadrature scheme has a common set of quadrature locations, but the quadrature weights are calculated on a species specific basis w_{ij} , where i is a sites index and j is a species index.

Fitting a model using quadratures generated from `ppmData`

Here we demonstrate how to fit a IPPM using `glm` function and a `ppmData` object. This approach is very similar to the method presented in Warton & Shepherd (2010), except we are using a quasi-random quadrature. We use `glm` to fit a IPPM for simplicity, but if one wanted to fit an IPPM using different statistical machinery, then one could – examples are: penalized regressions using `glmnet` (Friedman, Hastie, & Tibshirani, 2010) to achieve a `maxent` analysis (Renner & Warton, 2013), generalized additive models (e.g. Wood, 2017) or statistical learning approaches (e.g. Hastie, Tibshirani, Friedman, & Friedman, 2009).

```
#load relevant libraries
library(ppmData)
library(terra) ## need this load rasters
#read the data into R
path <- system.file("extdata", package = "ppmData")
lst <- list.files(path=path,pattern='*.tif',full.names = TRUE)
preds <- rast(lst)
#Format the data (subset and scale)
presences <- subset(snails,SpeciesID %in% "Tasmaphena sinclairi")
preds <- scale(preds)
#Obtain quadrature points, variables and weights
ppmdata <- ppmData(npoints = 20000, presences = presences,
                  window = preds[[1]], covariates = preds)
```

Once we have setup a `ppmData` object we can fit an inhomogeneous Poisson point process. Here is an example code for fitting it using `glm` from the R `stats` package (R Core Team, 2013). We just need to extract the data.frame from the `ppmData` object. This data.frame will contain all the needed information to fit an inhomogeneous Poisson point process. We also need to specify a formula for the model, the only trick is making the response equal to `presence/weights` (the expectation). The right side of the formula will represent the covariate function forms in the model. In this example we specify independent 2nd degree polynomials for each covariate, except for the spatial coordinates X & Y which we include in the model as interacting 2nd degree polynomials. Once this is done, we declare the weights in the `glm` function call and we now can fit a IPPM using `ppmData` and `glm`.

```
ppp <- ppmdata$ppmData
form <- presence/weights ~ poly(X,Y, degree = 2) +
                        poly(max_temp_hottest_month, degree = 2) +
                        poly(annual_mean_precip, degree = 2) +
                        poly(annual_mean_temp, degree = 2) +
                        poly(distance_from_main_roads, degree = 2)

ft.ppm <- glm(formula = form, data = ppp,
             weights = as.numeric(ppp$weights),
```

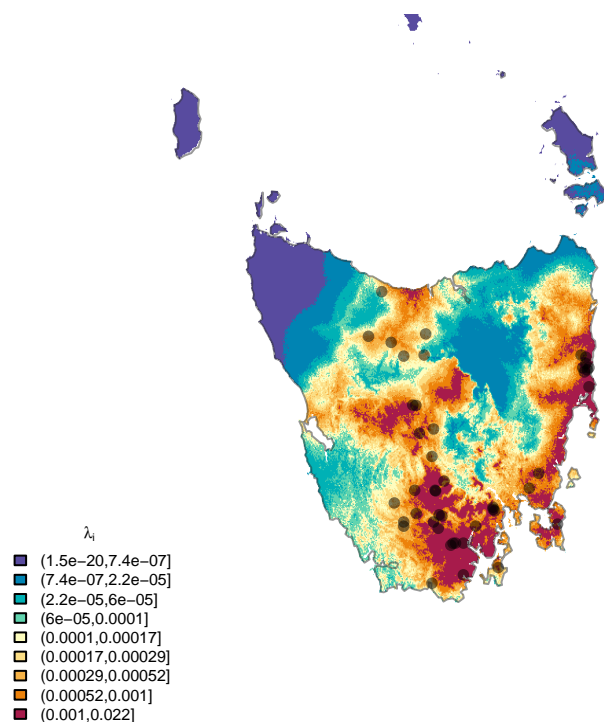


Figure 3: Predicted intensity per for $\approx 0.6 \text{ km}^2$ sized raster cell for the snail species *Tasmaphena sinclairi*. The points are the recorded presences of *Tasmaphena sinclairi* in the region.

```
family = poisson()
```

For the purpose of demonstration, we show how to predict the model using `terra` and a `glm` object. This will return the expected intensity if the areal weights of each were equal to 1. We can re-scale the intensity based on the resolution of the raster cells, resulting in the approximate intensity of the species per cell. This essentially gives us the expected count of presences per raster cell. See Figure 3 for example output.

Future work

One common problem when developing a quadrature is identifying the right number of quadrature points and identify locations which contribute the most information to log-likelihood. A common approach is to choose an arbitrary large number of quadrature locations (Renner et al., 2015) or fit multiple models at different grid resolutions to assess log-likelihood convergence (Warton & Shepherd, 2010). One way to improve computation efficiency of quadrature schemes would be to use inclusion probabilities (Foster et al., 2017). There will be regions which do not contribute very much to the log-likelihood (Warton & Aarts, 2013). These regions could be give sparser quasi-random samples, while areas which are likely to contribute larger amounts of information could be targeted. For

our future work, we envisage an adaptive scheme, where we start with a relatively small number of quadrature points and update the scheme in regions of the spatial domain $\mathcal{A} \in \mathbb{R}^2$ where the gradient of the integral is highly variable. Inclusion probabilities can be thus updated to increase the probability of a quasi-random sample in regions where the integral gradient (or other appropriate metrics) are complex. An adaptive approach would hone a quadrature scheme to improve computation efficient and target areas that will contribute more information to the log-likelihood.

Acknowledgments

We thank Piers Dunstan for comments on an earlier draft of this manuscript.

References

- Baddeley, A., Rubak, E., & Turner, R. (2015). *Spatial Point Patterns: Methodology and Applications with R*. London: Chapman; Hall/CRC Press. Retrieved from <https://www.routledge.com/Spatial-Point-Patterns-Methodology-and-Applications-with-R/Baddeley-Rubak-Turner/9781482210200/>
- Baddeley, A., & Turner, R. (2000). Practical maximum pseudolikelihood for spatial point patterns. *Australian & New Zealand Journal of Statistics*, 42(3), 283–322. doi:[10.1111/1467-842X.00128](https://doi.org/10.1111/1467-842X.00128)
- Berman, M., & Turner, T. R. (1992). Approximating point process likelihoods with GLIM. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 41(1), 31–38.
- Cressie, N. (1993). *Statistics for spatial data*. John Wiley & Sons.
- Diggle, P. J., Menezes, R., & Su, T. (2010). Geostatistical inference under preferential sampling. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 59(2), 191–232.
- Foster, S. D., Hosack, G. R., Lawrence, E., Przeslawski, R., Hedge, P., Caley, M. J., Barrett, N. S., et al. (2017). Spatially balanced designs that incorporate legacy sites. *Methods in Ecology and Evolution*, 8(11), 1433–1442.
- Friedman, J., Hastie, T., & Tibshirani, R. (2010). Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of Statistical Software*, 33(1), 1–22. doi:[10.18637/jss.v033.i01](https://doi.org/10.18637/jss.v033.i01)
- Grafström, A., & Tillé, Y. (2013). Doubly balanced spatial sampling with spreading and restitution of auxiliary totals. *Environmetrics*, 24(2), 120–131.
- Halton, J. H. (1960). On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numerische Mathematik*, 2(1), 84–90.
- Hastie, T., Tibshirani, R., Friedman, J. H., & Friedman, J. H. (2009). *The elements of statistical learning: Data mining, inference, and prediction* (Vol. 2). Springer.
- Hijmans, R. J. (2021). Terra: Spatial Data Analysis. R Package Version 1.1-4.
- Liu, J., & Vanhatalo, J. (2020). Bayesian model based spatiotemporal survey designs and partially observed log Gaussian Cox process. *Spatial statistics*, 35, 100392.
- Ovaskainen, O., & Soininen, J. (2011). Making more out of sparse data: Hierarchical modeling of species communities. *Ecology*, 92(2), 289–295. doi:[10.1890/10-1251.1](https://doi.org/10.1890/10-1251.1)
- Pebesma, E. (2018). Simple Features for R: Standardized Support for Spatial Vector Data. *The R Journal*, 10(1), 439–446. doi:[10.32614/RJ-2018-009](https://doi.org/10.32614/RJ-2018-009)

- Phillips, S. J., Dudík, M., Elith, J., Graham, C. H., Lehmann, A., Leathwick, J., & Ferrier, S. (2009). Sample selection bias and presence-only distribution models: Implications for background and pseudo-absence data. *Ecological Applications*, 19(1), 181–197. doi:[10.1890/07-2153.1](https://doi.org/10.1890/07-2153.1)
- R Core Team. (2013). *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. Retrieved from <http://www.R-project.org/>
- Renner, I. W., Elith, J., Baddeley, A., Fithian, W., Hastie, T., Phillips, S. J., Popovic, G., et al. (2015). Point process models for presence-only analysis. *Methods in Ecology and Evolution*, 6(4), 366–379. doi:[10.1111/2041-210X.12352](https://doi.org/10.1111/2041-210X.12352)
- Renner, I. W., & Warton, D. I. (2013). Equivalence of MAXENT and Poisson point process models for species distribution modeling in ecology. *Biometrics*, 69(1), 274–281.
- Robertson, B., Brown, J., McDonald, T., & Jaksons, P. (2013). BAS: Balanced acceptance sampling of natural resources. *Biometrics*, 69(3), 776–784.
- Sinclair, D. (2016). S-hull: A fast radial sweep-hull routine for Delaunay triangulation. *arXiv preprint arXiv:1604.01428*.
- Warton, D., & Aarts, G. (2013). Advancing our thinking in presence-only and used-available analysis. *Journal of Animal Ecology*, 82(6), 1125–1134.
- Warton, D., & Shepherd, L. (2010). Poisson point process models solve the "pseudo-absence problem" for presence-only data in ecology. *Annals of Applied Statistics*, 4(3), 1383–1402. doi:[10.1214/10-AOAS331](https://doi.org/10.1214/10-AOAS331)
- Wood, S. N. (2017). *Generalized Additive Models: An Introduction with R* (2nd ed.). Chapman; Hall/CRC.
- Woolley, S., Foster, S., & Dunstan, P. (2022). Ecomix: Finite mixture models for multiple species grouping of ecological data. Retrieved from <https://github.com/skiptoniam/ecomix>