Package ‘GRaF’

February 19, 2015

Type Package

Title Species distribution modelling using latent Gaussian random fields

Version 0.1-12

Date 2014-01-08

Author Nick Golding

Maintainer Nick Golding <nick.golding@linacre.ox.ac.uk>

Description Functions to fit, visualise and compare Gaussian random field species distribution models.

License GPL (>= 2)

Depends dismo

NeedsCompilation no

Repository CRAN

Date/Publication 2014-01-08 12:53:19

R topics documented:

GRaF-package .............................................................. 2
DIC ................................................................. 3
graf ............................................................... 4
plot.graf ......................................................... 6
plot3d .......................................................... 7
predict.graf ................................................... 8
print.graf ....................................................... 9

Index 11
Species distribution modelling using latent Gaussian random fields

Description

GRaF fits Gaussian random field species distribution models using Laplace approximations.

Details

<table>
<thead>
<tr>
<th>Package</th>
<th>GRaF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Package</td>
</tr>
<tr>
<td>Version</td>
<td>0.1-12</td>
</tr>
<tr>
<td>Date</td>
<td>2014-01-08</td>
</tr>
<tr>
<td>License</td>
<td>GPL (&gt;= 2)</td>
</tr>
</tbody>
</table>

GRaF provides functions to fit, visualise, and predict from Gaussian random field models for binary data, with an emphasis on species distribution models. The principle function is `graf` which returns a `graf` object, which can be manipulated using `plot`, `print` and `predict` methods among others.

GRaF is under active development and future releases will enable a range of additional methods, which may (or may not) include: models for count data, spatially and temporally explicit models, the ability to limit interactions between covariates, multi-species models and input and output of raster imagery. Please feel free to contact the maintainer with bug reports and suggestions!

The development version of this package is hosted on github (https://github.com/goldingn/GRaF) and a brief tutorial is available here: https://raw.githubusercontent.com/goldingn/intecol2013/master/tutorial/graf_workshop.html

Author(s)

Nick Golding <nick.golding@linacre.ox.ac.uk>

References


See Also

graf
DIC

Deviance information criterion for a graf model

Description

Calculates the deviance information criterion (DIC) and effective parameters for a graf model.

Usage

DIC(object)

Arguments

object A graf object.

Details

Information criteria can be used to compare the goodness of fit of two models to the same dataset whilst accounting for model complexity. Commonly used methods such as AIC are difficult to implement for models such a GRaF where the number of parameters in the model is not obvious. Spiegelhalter et al. (2002) proposed the effective number of parameters for these models and the associated DIC for model comparison. Here we calculate DIC by numerical integration. If weights are included in the model fitting they are taken into account when calculating the DIC. DIC should therefore not be used to compare between models with different observation weights.

Value

A vector containing:

- **DIC** The deviance information criterion.
- **pD** The estimated number of effective parameters in the model.

References


See Also

graf
**graf**

*Fit a GRaF model to binary data*

### Description

Sets up and fits a GRaF model, returning an object of class `graf` for which functions are available to visualise, compare and make predictions from models.

### Usage

```r
graf(y, x, error = NULL, weights = NULL, prior = NULL, l = NULL, opt.l = FALSE,
theta.prior.pars = c(log(10), 1), hessian = FALSE, opt.control = list(),
verbose = FALSE, method = c('Laplace', 'EP'))
```

### Arguments

- **y**: A vector denoting presence (1) and absence (0) records.
- **x**: A dataframe of covariates which may contain factors as well as continuous variables. Factor covariates can still interact with other covariates, but they are not influenced by either the mean function or the length-scale parameters and therefore give a maximum-likelihood estimate.
- **error**: An optional matrix of standard deviations associated with x. If this is missing, covariates are assumed to be measured without error.
- **weights**: An optional vector of weights to be used in the fitting process. Weights must be positive or zero. Observations with a weight of zero will be ignored.
- **prior**: An optional R function providing an *a priori* estimate of the probability of presence of the species given the covariates. The function must take a dataframe of environmental covariates (matching x) as input and return a corresponding vector of the probability of presence. If NULL a flat prior is used which gives the species prevalence as the probability of presence at all sites.
- **l**: An optional vector of lengthscales for each dimension of the Gaussian field. The lengthscale controls the smoothness of the fitted function, with a higher value giving a flatter curve. If NULL lengthscales are approximated as 8 times the ratio between the standard deviation of environmental variables at presence sites and the standard deviation of environmental variables at all sites. If `opt.l` is specified it gives the starting position for the optimisation algorithm. Lengthscales specified for factors will be ignored.
- **opt.l**: Whether to optimise the lengthscale parameters with respect to the model marginal likelihood using `optim` (with method BFGS) and return a model fitted with the maximum *a posteriori* estimate of the length scales. With a large number of data points this can be quite time consuming.
- **theta.prior.pars**: If `opt.l` = `TRUE`, allows the user to specify the mean and standard deviation of the normal hyperprior over the log-lengthscale hyperparameter `theta`. By default the mean is set at `log(10)` and the standard deviation at 1.
**hessian**

If `opt.l = TRUE`, whether to calculate the hessian matrix whilst optimising lengthscales. Calculating the hessian is more time consuming and is not recommended for general use, but advanced users could use it to approximate the full posterior density of the lengthscale hyperparameters.

**opt.control**

If `opt.l = TRUE`, allows the user to pass a list of options directly to the `control` argument of the `optim` function used to carry out the lengthscale optimisation. For large models, adjusting control parameters such as the convergence tolerance (`reltol`) can reduce computation time, with an associated loss of precision in the optimisation. See `control` in `optim` for more details.

**verbose**

Logical, whether to display (limited) progress information for the fitting algorithms.

**method**

The approximation method used to fit model. Currently only 'Laplace' (Laplace approximation - the default) and 'EP' (the expectation-propagation algorithm) are available. Whilst the EP algorithm is known to provide a more accurate approximation than Laplace under certain circumstances (see e.g. section 3.7.2 of Rasmussen & Williams (2006)), it has the disadvantages of much greater computational cost and no method for incorporating regression weights (at least in this implementation). In practice, models fitted using the two methods are likely to be very similar. The two approximations are described in detail by Rasmussen & Williams (2006).

### Details

In addition to the dataframe of covariates `x` a matrix of corresponding standard deviations, describing the measurement error, may be specified via the `error` argument.

If prior knowledge about the species’ ecology is available (e.g. environmental thresholds beyond which the species cannot persist), this may be incorporated into the GRaF model by providing an R function via the `prior` argument.

The smoothness of the fitted random field is controlled by a vector of lengthscales which is used to parameterise a squared exponential covariance function, with one lengthscale for each covariate. `graf` centres and standardises covariates before fitting and the lengthscales therefore relate to these scaled covariates. Lengthscales can be specified by the user via the `l` argument, roughly approximated by `graf` by setting `l = NULL` or optimised by setting `opt.l = TRUE`.

### Value

A `graf` object which can be viewed and manipulated using `print`, `plot` and `predict` functions.

### References


See Also

print.graf, plot.graf, predict.graf, plot3d, DIC, optim.

Examples

# load Anguilla data from the dismo package
data(anguilla_train)

# get the first 100 presence-absence records
y <- anguilla_train$Angaus[1:100]
# get covariates, removing LocSed (contains NAs) and make DSDam a factor
x <- anguilla_train[1:100, 3:13]
x$DSDam <- factor(x$DSDam)

# fit a presence-absence model to the data
m1 <- graf(y, x)

# print a brief summary of the model
m1 # print(m1)

# plot the (conditional) effect of each term
par(mfrow = c(3, 4))
plot(m1)

# visualise the interaction between SegTSeas and SegLowFlow
par(mfrow = c(2, 2), mar = rep(2, 4))
plot3d(m1, dims = c(2, 3), theta = -110)

# fit another model with flatter responses
m2 <- graf(y, x, 1 = m1$ls * 2)

# compare the two models by DIC
DIC(m1)
DIC(m2)

# m2 has lower DIC so could be considered preferable

# predict back to dataset
pred <- predict(m2, x)
head(pred)

---

plot.graf | Plot graf models

Description

plot.graf provides basic plots to visualise fitted terms in graf models. The predicted probability of presence along each covariate is plotted, along with associate credible intervals, with other covariates are held either at their average value or near the peak (point of highest probability) of the latent random field.
plot3d

Usage

```r
## S3 method for class 'graf'
plot(x, vars = NULL, resolution = 50, CI = 0.95, prior = FALSE, data = TRUE, jitter = 1,
     peak = FALSE, ...)```

Arguments

- `x`: A `graf` object.
- `vars`: A vector giving the column index of covariates for which to plot fitted terms. If `NULL` fitted terms are plotted for all covariates.
- `resolution`: The resolution of the prediction line for each covariate.
- `CI`: The level at which to calculate predictive credible intervals. The default value returns upper and lower 95% credible intervals.
- `prior`: Whether to additionally plot the value of the prior function.
- `data`: Whether to add rug plots of the training data.
- `jitter`: The amount to jitter the observed data points (only applies if `data = TRUE`). Set to 0 for no jittering.
- `peak`: Whether to hold other covariates at the value of the training datapoint with the highest probability of presence. If `FALSE` covariates are held at their mean value in the training data set.
- `...`: Additional arguments to be passed to `plot`.

Value

A series of plots (one per covariate) of the marginal effect of covariates (near the peak) on the probability of species presence.

See Also

`graf`, `plot3d`

---

**Description**

`plot3d` provides basic surface plots to visualise interactions between fitted terms in `graf` models. The predicted probability of presence along two covariates is plotted with other covariates held either at their average value or near the peak (point of highest probability) of the latent random field. The prior function and credible intervals may also be plotted. 3d plots for factors are not implemented.
Usage

`plot3d(object, dims = c(1, 2), resolution = 20, CI = 0.95, prior = TRUE, CIs = TRUE, theta = 30, phi = 30, peak = FALSE, ...)`

Arguments

- `object`: A `graf` object.
- `dims`: A vector of length 2 giving the column numbers of the covariates to plot.
- `resolution`: The resolution of the prediction surface along each covariate.
- `CI`: The level at which to calculate predictive credible intervals. The default value returns upper and lower 95% credible intervals.
- `prior`: Whether to additionally plot the value of the prior function.
- `CIs`: Whether to additionally plot the credible intervals.
- `theta`: Viewing angle for `persp`, theta gives the azimuthal direction.
- `phi`: Viewing angle for `persp`, phi gives the colatitude.
- `peak`: Whether to hold other covariates at the value of the training datapoint with the highest probability of presence. If FALSE covariates are held at their mean value in the training data set.
- `...`: Additional arguments to pass to `persp`.

Value

One or more surface plots of the marginal effect of a pair of predictors on the probability of species presence.

See Also

`graf`, `plot.graf`  

`predict.graf` Predictions for `graf` objects.

Description

Predictions and associated credible intervals for `graf` objects, either for the data used for fitting or for a different dataset.

Usage

```r
## S3 method for class 'graf'
predict(object, newdata = NULL, type = c("response", "latent"), CI = 0.95, maxn = NULL, ...)```
Arguments

object  A graf object.
newdata  An optional dataframe giving covariates to predict to. If NULL a prediction for the covariates used to fit the model is returned.
type  The level of the prediction. "response" (default) gives predictions on the probability scale and "latent" gives predictions on the scale of the latent Gaussian.
CI  The level at which to calculate predictive credible intervals. The default value returns upper and lower 95% credible intervals. If credible intervals are not required this can be set to NULL. If type = 'latent' setting CI = 'std' returns the mean and standard deviation of the latent field.
maxn  The maximum number of records to predict to in each batch. To avoid computationally expensive operations on large matrices, predict.graf splits the dataset for prediction into batches with maximum number of records maxn. This can be adjusted by the user to optimise computational efficiency on different machines. If maxn = NULL maxn is set at approximately one tenth the number rows used to fit the model.
...  Additional arguments for future versions.

Value

A matrix of posterior modes and optionally credible intervals.

See Also

graf

print.graf  Print a graf object.

Description

Prints a brief summary of a graf object.

Usage

## S3 method for class 'graf'
print(x, ...)

Arguments

x  A graf object.
...  Additional arguments to pass to print

Value

The number of presence and absence/background records and covariates used to fit the model.
See Also

graf
Index

*Topic package
   GRAF-package, 2

DIC, 3, 6

GRAF (GRAF-package), 2
graf, 2, 3, 4, 7–10
GRAF-package, 2

optim, 4–6

plot.graf, 6, 6, 8
plot3d, 6, 7, 7
predict.graf, 6, 8
print.graf, 6, 9