Package 'MRFcov'

March 18, 2021

Type Package

Title Markov Random Fields with Additional Covariates

Version 1.0.38

Date/Publication 2021-03-18 06:40:03 UTC

URL https://github.com/nicholasjclark/MRFcov

Maintainer Nicholas J Clark <nicholas.j.clark1214@gmail.com>

Description Approximate node interaction parameters of Markov Random Fields graphical networks. Models can incorporate additional covariates, allowing users to estimate how interactions between nodes in the graph are predicted to change across covariate gradients. The general methods implemented in this package are described in Clark et al. (2018) <doi:10.1002/ecy.2221>.

Depends R (>= 3.3.3), glmnet

Imports purrr, parallel, plyr, ggplot2, dplyr, caret, gridExtra, magrittr, Matrix, stats, utils, reshape2, sfsmisc, igraph, grDevices, pbapply, mgcv, MASS

License GPL-3

Encoding UTF-8

RoxygenNote 7.1.1

LazyData true

Suggests testthat, knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Nicholas J Clark [aut, cre],
Konstans Wells [aut],
Oscar Lindberg [aut]

Repository CRAN

R topics documented:

Bird.parasites 2
bootstrap_MRF 3
cv_MRF_diag 5
MRFcov 9
MRFcov_spatial 12
plotMRF_hm 14
predict_MRF 15
predict_MRFnetworks 17
prep_MRF_covariates 19
prep_MRF_covariates_spatial 19

Index **21**

`Bird.parasites` *Blood parasite occurrences in New Caledonian birds.*

Description

A dataset containing binary occurrences of four blood parasite species in New Caledonian birds. The first four variables represent the parasite occurrences and the last variable is a scaled continuous covariate representing host relative abundance.

Usage

```
Bird.parasites
```

Format

A data frame with 449 rows and 5 variables:

Hzosteropis binary occurrence of *Haemoproteus zosteropis*

Hkillangoi binary occurrence of *Haemoproteus killangoi*

Plas binary occurrence of *Plasmdodium* species

Microfilaria binary occurrence of Microfilaria species

scale.prop.zos scaled numeric variable representing relative abundance of *Zosterops* host species

Source

doi: 10.5061/dryad.pp6k4

References

Clark, N.J., Wells, K., Dimitrov, D. & Clegg, S.M. (2016) Co-infections and environmental conditions drive the distributions of blood parasites in wild birds. Journal of Animal Ecology, 85, 1461-1470.

bootstrap_MRF *Bootstrap observations to estimate MRF parameter coefficients*

Description

This function runs MRFcov models multiple times to capture uncertainty in parameter esimates. The dataset is shuffled and missing values (if found) are imputed in each bootstrap iteration.

Usage

```
bootstrap_MRF(
  data,
  n_bootstraps,
  sample_seed,
  symmetrise,
  n_nodes,
  n_cores,
  n_covariates,
  family,
  sample_prop,
  spatial = FALSE,
  coords = NULL
)
```

Arguments

data	Dataframe. The input data where the n_nodes left-most variables are variables that are to be represented by nodes in the graph. Note that NA's are allowed for covariates. If present, these missing values will be imputed from the distribution rnorm(mean = 0, sd = 1), which assumes that all covariates are scaled and centred (i.e. by using the function scale or similar)
n_bootstraps	Positive integer. Represents the total number of bootstrap samples to test. Default is 100.
sample_seed	Numeric. Used as the seed value for generating bootstrap replicates, allowing users to generate replicated datasets on different systems. Default is a random seed
symmetrise	The method to use for symmetrising corresponding parameter estimates (which are taken from separate regressions). Options are min (take the coefficient with the smallest absolute value), max (take the coefficient with the largest absolute value) or mean (take the mean of the two coefficients). Default is mean
n_nodes	Positive integer. The index of the last column in data which is represented by a node in the final graph. Columns with index greater than n_nodes are taken as covariates. Default is the number of columns in data, corresponding to no additional covariates
n_cores	Integer. The number of cores to spread the job across using makePSOCKcluster. Default is 1 (no parallelisation)

n_covariates	Positive integer. The number of covariates in data, before cross-multiplication. Default is ncol(data) -n_nodes
family	The response type. Responses can be quantitative continuous (family = "gaussian"), non-negative counts (family = "poisson") or binomial 1s and 0s (family = "binomial")
sample_prop	Positive probability value indicating the proportion of rows to sample from data in each bootstrap iteration. Default is no subsampling (sample_prop == 1)
spatial	Logical. If TRUE, spatial MRF / CRF models are bootstrapped using MRFcov_spatial. Note, GPS coordinates must be supplied as coords for spatial models to be run. Smoothed spatial splines will be included in each node-wise regression as covariates. This ensures resulting node interaction parameters are estimated after accounting for possible spatial autocorrelation. Note that interpretation of spatial autocorrelation is difficult, and so it is recommended to compare predictive capacities spatial and non-spatial CRFs through the predict_MRF function
coords	A two-column dataframe (with nrow(coords) == nrow(data)) representing the spatial coordinates of each observation in data. Ideally, these coordinates will represent Latitude and Longitude GPS points for each observation.

Details

MRFcov models are fit via cross-validation using cv.glmnet. For each model, the data is bootstrapped by shuffling row observations and fitting models to a subset of observations to account for uncertainty in parameter estimates. Parameter estimates from the set of bootstrapped models are summarised to present means and confidence intervals (as 95 percent quantiles).

Value

A list containing:

- direct_coef_means: dataframe containing mean coefficient values taken from all bootstrapped models across the iterations
- direct_coef_upper90 and direct_coef_lower90: dataframes containing coefficient 95 percent and 5 percent quantiles taken from all bootstrapped models across the iterations
- indirect_coef_mean: list of symmetric matrices (one matrix for each covariate) containing mean effects of covariates on pairwise interactions
- mean_key_coefs: list of matrices of length n_nodes containing mean covariate coefficient values and their relative importances (using the formula x^2 / sum(x^2) taken from all bootstrapped models across iterations. Only coefficients with mean relative importances >0.01 are returned. Note, relative importance are only useful if all covariates are on a similar scale.
- mod_type: A character stating the type of model that was fit (used in other functions)
- mod_family: A character stating the family of model that was fit (used in other functions)
- poiss_sc_factors: A vector of the square-root mean scaling factors used to standardise poisson variables (only returned if family = "poisson")

See Also

MRFcov, MRFcov_spatial, cv.glmnet

Examples

```
data("Bird.parasites")

# Perform 2 quick bootstrap replicates using 70% of observations
bootedCRF <- bootstrap_MRF(data = Bird.parasites,
                           n_nodes = 4,
                           family = 'binomial',
                           sample_prop = 0.7,
                           n_bootstraps = 2)


# Small example of using spatial coordinates for a spatial CRF
Latitude <- sample(seq(120, 140, length.out = 100), nrow(Bird.parasites), TRUE)
Longitude <- sample(seq(-19, -22, length.out = 100), nrow(Bird.parasites), TRUE)
coords <- data.frame(Latitude = Latitude, Longitude = Longitude)
bootedSpatial <- bootstrap_MRF(data = Bird.parasites, n_nodes = 4,
                               family = 'binomial',
                               spatial = TRUE,
                               coords = coords,
                               sample_prop = 0.5,
                               n_bootstraps = 2)
```

---

cv_MRF_diag *MRF cross validation and assessment of predictive performance*

Description

cv_MRF_diag runs cross validation of MRFcov models and tests predictive performance.

cv_MRF_diag_rep fits a single node-optimised model and test's this model's predictive performance across multiple test subsets of the data.

cv_MRF_diag_rep_spatial fits a single node-optimised spatial model and test's this model's predictive performance across multiple test subsets of the data.

All cv_MRF functions assess model predictive performance and produce either diagnostic plots or matrices of predictive metrics.

Usage

```
cv_MRF_diag(
  data,
  symmetrise,
  n_nodes,
  n_cores,
  sample_seed,
  n_folds,
  n_fold_runs,
  n_covariates,
```

```
  compare_null,
  family,
  plot = TRUE,
  cached_model,
  cached_predictions,
  mod_labels = NULL
)

cv_MRF_diag_rep(
  data,
  symmetrise,
  n_nodes,
  n_cores,
  sample_seed,
  n_folds,
  n_fold_runs,
  n_covariates,
  compare_null,
  family,
  plot = TRUE
)

cv_MRF_diag_rep_spatial(
  data,
  coords,
  symmetrise,
  n_nodes,
  n_cores,
  sample_seed,
  n_folds,
  n_fold_runs,
  n_covariates,
  compare_null,
  family,
  plot = TRUE
)
```

Arguments

`data`	Dataframe. The input data where the `n_nodes` left-most variables are variables that are to be represented by nodes in the graph. Note that `NA`'s are allowed for covariates. If present, these missing values will be imputed from the distribution `rnorm(mean = 0, sd = 1)`, which assumes that all covariates are scaled and centred (i.e. by using the function `scale` or similar)
`symmetrise`	The method to use for symmetrising corresponding parameter estimates (which are taken from separate regressions). Options are `min` (take the coefficient with the smallest absolute value), `max` (take the coefficient with the largest absolute value) or `mean` (take the mean of the two coefficients). Default is `mean`

n_nodes — Positive integer. The index of the last column in data which is represented by a node in the final graph. Columns with index greater than n_nodes are taken as covariates. Default is the number of columns in data, corresponding to no additional covariates

n_cores — Positive integer. The number of cores to spread the job across using makePSOCKcluster. Default is 1 (no parallelisation)

sample_seed — Numeric. This seed will be used as the basis for dividing data into folds. Default is a random seed between 1 and 100000

n_folds — Integer. The number of folds for cross-validation. Default is 10

n_fold_runs — Integer. The number of total training runs to perform. During each run, the data will be split into n_folds folds and the observed data in each fold will be compared to their respective predictions. Defaults to n_folds

n_covariates — Positive integer. The number of covariates in data, before cross-multiplication

compare_null — Logical. If TRUE, null models will also be run and plotted to assess the influence of including covariates on model predictive performance. Default is FALSE

family — The response type. Responses can be quantitative continuous (family = "gaussian"), non-negative counts (family = "poisson") or binomial 1s and 0s (family = "binomial").

plot — Logical. If TRUE, ggplot2 objects are returned. If FALSE, the prediction metrics are returned as a matrix. Default is TRUE

cached_model — Used by function cv_MRF_diag_rep to store an optimised model and prevent unneccessary replication of node-optimised model fitting

cached_predictions — Used by function cv_MRF_diag_rep to store predictions from optimised models and prevent unneccessary replication

mod_labels — Optional character string of labels for the two models being compared (if compare_null == TRUE)

coords — A two-column dataframe (with nrow(coords) == nrow(data)) representing the spatial coordinates of each observation in data. Ideally, these coordinates will represent Latitude and Longitude GPS points for each observation.

Details

Node-optimised models are fitted using cv.glmnet, and these models is used to predict data test subsets. Test and training data subsets are created using createFolds.

To account for uncertainty in parameter estimates and in random fold generation, it is recommended to perform cross-validation multiple times (by controlling the n_fold_runs argument) using cv_MRF_diag_rep to supply a single cached model and that model's predictions. This is useful for optimising a single model (using cv.glmnet) and testing this model's predictive performance across many test subsets. Alternatively, one can run cv_MRF_diag many times to fit different models in each iteration. This will be slower but technically more sound

Value

If `plot = TRUE`, a `ggplot2` object is returned. This will be a plot containing boxplots of predictive metrics across test sets using the optimised model (see `cv.glmnet` for further details of `lambda1` optimisation). If `plot = FALSE`, a matrix of prediction metrics is returned.

References

Clark, NJ, Wells, K and Lindberg, O. Unravelling changing interspecific interactions across environmental gradients using Markov random fields. (2018). Ecology doi: 10.1002/ecy.2221 Full text here.

See Also

`MRFcov`, `predict_MRF`, `cv.glmnet`

Examples

```
data("Bird.parasites")
# Generate boxplots of model predictive metrics
cv_MRF_diag(data = Bird.parasites, n_nodes = 4,
           n_cores = 3, family = 'binomial')

# Generate boxplots comparing the CRF to an MRF model (no covariates)
cv_MRF_diag(data = Bird.parasites, n_nodes = 4,
           n_cores = 3, family = 'binomial',
           compare_null = TRUE)

# Replicate 10-fold cross-validation 100 times
cv.preds <- cv_MRF_diag_rep(data = Bird.parasites, n_nodes = 4,
                            n_cores = 3, family = 'binomial',
                            compare_null = TRUE,
                            plot = FALSE, n_fold_runs = 100)

# Plot model sensitivity and % true predictions
library(ggplot2)
gridExtra::grid.arrange(
 ggplot(data = cv.preds, aes(y = mean_sensitivity, x = model)) +
       geom_boxplot() + theme(axis.text.x = ggplot2::element_blank()) +
       labs(x = ''),
 ggplot(data = cv.preds, aes(y = mean_tot_pred, x = model)) +
       geom_boxplot(),
       ncol = 1,
 heights = c(1, 1))

# Create some sample Poisson data with strong correlations
cov <- rnorm(500, 0.2)
cov2 <- rnorm(500, 4)
sp.2 <- ceiling(rnorm(500, 1)) + (cov * 2)
sp.2[sp.2 < 0] <- 0
poiss.dat <- data.frame(sp.1 = ceiling(rnorm(500, 1) + cov2 * 1.5),
```

```
                        sp.2 = sp.2, sp.3 = (sp.2 * 2) + ceiling(rnorm(500, 0.1)))
poiss.dat[poiss.dat < 0] <- 0
poiss.dat$cov <- cov
poiss.dat$cov2 <- cov2

# A CRF should produce a better fit (lower deviance, lower MSE)
cvMRF.poiss <- cv_MRF_diag(data = poiss.dat, n_nodes = 3,
                           n_folds = 10,
                           family = 'poisson',
                           compare_null = TRUE, plot = TRUE)
```

MRFcov *Markov Random Fields with covariates*

Description

This function is the workhorse of the MRFcov package, running separate penalized regressions for each node to estimate parameters of Markov Random Fields (MRF) graphs. Covariates can be included (a class of models known as Conditional Random Fields; CRF), to estimate how interactions between nodes vary across covariate magnitudes.

Usage

```
MRFcov(
  data,
  symmetrise,
  prep_covariates,
  n_nodes,
  n_cores,
  n_covariates,
  family,
  bootstrap = FALSE,
  progress_bar = FALSE
)
```

Arguments

data	A dataframe. The input data where the n_nodes left-most variables are variables that are to be represented by nodes in the graph
symmetrise	The method to use for symmetrising corresponding parameter estimates (which are taken from separate regressions). Options are min (take the coefficient with the smallest absolute value), max (take the coefficient with the largest absolute value) or mean (take the mean of the two coefficients). Default is mean
prep_covariates	Logical. If TRUE, covariate columns will be cross-multiplied with nodes to prep the dataset for MRF models. Note this is only useful when additional covariates

	are provided. Therefore, if n_nodes < ncol(data), default is TRUE. Otherwise, default is FALSE. See prep_MRF_covariates for more information
n_nodes	Positive integer. The index of the last column in data which is represented by a node in the final graph. Columns with index greater than n_nodes are taken as covariates. Default is the number of columns in data, corresponding to no additional covariates
n_cores	Positive integer. The number of cores to spread the job across using makePSOCKcluster. Default is 1 (no parallelisation)
n_covariates	Positive integer. The number of covariates in data, before cross-multiplication. Default is ncol(data) -n_nodes
family	The response type. Responses can be quantitative continuous (family = "gaussian"), non-negative counts (family = "poisson") or binomial 1s and 0s (family = "binomial"). If using (family = "binomial"), please note that if nodes occur in less than 5 percent of observations this can make it generally difficult to estimate occurrence probabilities (on the extreme end, this can result in intercept-only models being fitted for the nodes in question). The function will issue a warning in this case. If nodes occur in more than 95 percent of observations, this will return an error as the cross-validation step will generally be unable to proceed. For family = 'poisson' models, all returned coefficients are estimated on the identity scale AFTER using a nonparanormal transformation. See vignette("Gaussian_Poisson_CRFs") for details of interpretation
bootstrap	Logical. Used by bootstrap_MRF to reduce memory usage
progress_bar	Logical. Progress bar in pbapply is used if TRUE, but this slows estimation.

Details

Separate penalized regressions are used to approximate MRF parameters, where the regression for node j includes an intercept and coefficients for the abundance (families gaussian or poisson) or presence-absence (family binomial) of all other nodes (/j) in data. If covariates are included, coefficients are also estimated for the effect of the covariate on j, and for the effects of the covariate on interactions between j and all other nodes (/j). Note that interaction coefficients must be estimated between variables that are on roughly the same scale, as the resulting parameter estimates are unified into a Markov Random Field using the specified symmetrise function. Counts for poisson variables, which are often not on the same scale, will therefore be normalised with a nonparanormal transformation x = qnorm(rank(log2(x + 0.01)) / (length(x) + 1)). These transformed counts will be used in a (family = "gaussian") model and their respective raw distribution parameters returned so that coefficients can be back-transformed for interpretation (this back-transformation is performed automatatically by other functions including predict_MRF and cv_MRF_diag). Gaussian variables are not automatically transformed, so if they cover quite different ranges and scales, then it is recommended to scale them prior to fitting models. For more information on this process, use vignette("Gaussian_Poisson_CRFs")

Note that since the number of parameters to estimate in each node-wise regression quickly increases with increasing numbers of nodes and covariates, LASSO penalization is used to regularize regressions. This is done by minimising the cross-validated mean error for each node separately using cv.glmnet. In this way, we maximise the log-likelihood of each node separately before unifying the nodes into a graph.

Value

A `list` containing:

- `graph`: Estimated parameter `matrix` of pairwise interaction effects
- `intercepts`: Estimated parameter `vector` of node intercepts
- `indirect_coefs`: `list` containing matrices representing indirect effects of each covariate on pairwise node interactions
- `direct_coefs`: `matrix` of direct effects of each parameter on each outcome node. For `family = 'binomial'` models, all coefficients are estimated on the logit scale.
- `param_names`: Character string of covariate parameter names
- `mod_type`: A character stating the type of model that was fit (used in other functions)
- `mod_family`: A character stating the family of model that was fit (used in other functions)
- `poiss_sc_factors`: A matrix of the estimated negative binomial or poisson parameters for each raw node variable (only returned if `family = "poisson"`). These are needed for converting coefficients back to their original distribution, and are used for prediction purposes only

References

Ising, E. (1925). Beitrag zur Theorie des Ferromagnetismus. Zeitschrift für Physik A Hadrons and Nuclei, 31, 253-258.

Cheng, J., Levina, E., Wang, P. & Zhu, J. (2014). A sparse Ising model with covariates. (2012). Biometrics, 70, 943-953.

Clark, NJ, Wells, K and Lindberg, O. Unravelling changing interspecific interactions across environmental gradients using Markov random fields. (2018). Ecology doi: 10.1002/ecy.2221 Full text here.

Sutton C, McCallum A. An introduction to conditional random fields. Foundations and Trends in Machine Learning 4, 267-373.

See Also

Cheng et al. (2014), Sutton & McCallum (2012) and Clark et al. (2018) for overviews of Conditional Random Fields. See `cv.glmnet` for details of cross-validated optimization using LASSO penalty. Worked examples to showcase this function can be found using `vignette("Bird_Parasite_CRF")` and `vignette("Gaussian_Poisson_CRFs")`

Examples

```
data("Bird.parasites")
CRFmod <- MRFcov(data = Bird.parasites, n_nodes = 4, family = 'binomial')
```

MRFcov_spatial *Spatially structured Markov Random Fields with covariates*

Description

This function calls the MRFcov function to fit separate penalized regressions for each node and approximate parameters of Markov Random Fields (MRF) graphs. Supplied GPS coordinates are used to account for spatial autocorrelation via Gaussian Process spatial regression splines.

Usage

```
MRFcov_spatial(
  data,
  symmetrise,
  prep_covariates,
  n_nodes,
  n_cores,
  n_covariates,
  family,
  coords,
  prep_splines = TRUE,
  bootstrap = FALSE,
  progress_bar = FALSE
)
```

Arguments

data	A dataframe. The input data where the n_nodes left-most variables are variables that are to be represented by nodes in the graph
symmetrise	The method to use for symmetrising corresponding parameter estimates (which are taken from separate regressions). Options are min (take the coefficient with the smallest absolute value), max (take the coefficient with the largest absolute value) or mean (take the mean of the two coefficients). Default is mean
prep_covariates	Logical. If TRUE, covariate columns will be cross-multiplied with nodes to prep the dataset for MRF models. Note this is only useful when additional covariates are provided. Therefore, if n_nodes < ncol(data), default is TRUE. Otherwise, default is FALSE. See prep_MRF_covariates for more information
n_nodes	Positive integer. The index of the last column in data which is represented by a node in the final graph. Columns with index greater than n_nodes are taken as covariates. Default is the number of columns in data, corresponding to no additional covariates
n_cores	Positive integer. The number of cores to spread the job across using makePSOCKcluster. Default is 1 (no parallelisation)
n_covariates	Positive integer. The number of covariates in data, before cross-multiplication. Default is ncol(data) -n_nodes

family — The response type. Responses can be quantitative continuous (family = "gaussian"), non-negative counts (family = "poisson") or binomial 1s and 0s (family = "binomial"). If using (family = "binomial"), please note that if nodes occur in less than 5 percent of observations this can make it generally difficult to estimate occurrence probabilities (on the extreme end, this can result in intercept-only models being fitted for the nodes in question). The function will issue a warning in this case. If nodes occur in more than 95 percent of observations, this will return an error as the cross-validation step will generally be unable to proceed. For family = 'poisson' models, all returned coefficients are estimated on the identity scale AFTER using a nonparanormal transformation. See vignette("Gaussian_Poisson_CRFs") for details of interpretation

coords — A two-column dataframe (with nrow(coords) == nrow(data)) representing the spatial coordinates of each observation in data. Ideally, these coordinates will represent Latitude and Longitude GPS points for each observation. The coordinates are used to create smoothed Gaussian Process spatial regression splines via smooth.construct2. Here, the basis dimension of the smoothed term is chosen based on the number of unique GPS coordinates in coords. If this number is less than 100, then this number is used. If the number of unique coordiantes is more than 100, a value of 100 is used (this parameter needs to be large in order to ensure enough degrees of freedom for estimating 'wiggliness' of the smooth term; see choose.k for details). These splines will be included in each node-wise regression as additional penalized covariates. This ensures that resulting node interaction parameters are estimated after accounting for possible spatial autocorrelation. Note that interpretation of spatial autocorrelation is difficult, and so it is recommended to compare predictive capacities spatial and non-spatial CRFs through the predict_MRF function

prep_splines — Logical. If spatial splines are already included in data, set to FALSE. Default is TRUE

bootstrap — Logical. Used by bootstrap_MRF to reduce memory usage

progress_bar — Logical. Progress bar in pbapply is used if TRUE, but this slows estimation.

Value

A list of all elements contained in a returned MRFcov object, with the inclusion of a dataframe called mrf_data. This contains all prepped covariates including the added spatial regression splines, and should be used as data when generating predictions via predict_MRF or predict_MRFnetworks

References

Kammann, E. E. and M.P. Wand (2003) Geoadditive Models. Applied Statistics 52(1):1-18.

See Also

See smooth.construct2 and smooth.construct.gp.smooth.spec for details of Gaussian process spatial regression splines. Worked examples to showcase this function can be found using vignette("Bird_Parasite_CRF")

Examples

```
data("Bird.parasites")
Latitude <- sample(seq(120, 140, length.out = 100), nrow(Bird.parasites), TRUE)
Longitude <- sample(seq(-19, -22, length.out = 100), nrow(Bird.parasites), TRUE)
coords <- data.frame(Latitude = Latitude, Longitude = Longitude)
CRFmod_spatial <- MRFcov_spatial(data = Bird.parasites, n_nodes = 4,
                                 family = 'binomial', coords = coords)
```

`plotMRF_hm` *Plot MRF interaction parameters as a heatmap*

Description

This function uses outputs from fitted `MRFcov` and `bootstrap_MRF` models to plot a heatmap of node interaction coefficients.

Usage

```
plotMRF_hm(MRF_mod, node_names, main, plot_observed_vals, data)
```

Arguments

`MRF_mod`	A fitted `MRFcov` or `bootstrap_MRF` object
`node_names`	A character vector of species names for axis labels. Default is to use rownames from the `MRFcov$graph` slot
`main`	An optional character title for the plot
`plot_observed_vals`	Logical. If `TRUE` and the family of the fitted `MRFcov` model is `'binomial'`, then raw observed occurrence and co-occurrence values will be extracted from `data` and overlaid on the resulting heatmap. Note, this option is not available for `bootstrap_MRF` models
`data`	Optional `dataframe` containing the input data where the left-most columns represent binary occurrences of species that are represented by nodes in the graph. This call is only necessary if users wish to overlay raw observed occurrence and co-occurrence values on the heatmap of node interaction coefficients (only avaiable for `family = 'binomial'` models)

Details

Interaction parameters from `MRF_mod` are plotted as a heatmap, where red colours indicate positive interactions and blue indicate negative interactions. If `plot_observed_vals == TRUE`, raw observed values of single occurrences (on the diagonal) and co-occurrences for each species in `data` are overlaid on the plot (only avaiable for `family = 'binomial'` models). Note, this option is not available for `bootstrap_MRF` models

Value

A ggplot2 object

See Also

MRFcov bootstrap_MRF

Examples

```
data("Bird.parasites")
CRFmod <- MRFcov(data = Bird.parasites, n_nodes = 4, family = 'binomial')
plotMRF_hm(MRF_mod = CRFmod)
plotMRF_hm(MRF_mod = CRFmod, plot_observed_vals = TRUE, data = Bird.parasites)

#To plot as an igraph network instead, we can simply extract the adjacency matrix
net <- igraph::graph.adjacency(CRFmod$graph, weighted = TRUE, mode = "undirected")
igraph::plot.igraph(net, layout = igraph::layout.circle,
                    edge.width = abs(igraph::E(net)$weight),
                    edge.color = ifelse(igraph::E(net)$weight < 0, 'blue', 'red'))
```

---

predict_MRF *Predict training observations from fitted MRFcov models*

Description

This function calculates linear predictors for node observations using coefficients from an MRFcov or MRFcov_spatial object.

Usage

```
predict_MRF(
  data,
  MRF_mod,
  prep_covariates = TRUE,
  n_cores,
  progress_bar = FALSE
)
```

Arguments

data	Dataframe. The input data to be predicted, where the n_nodes left-most variables are are variables that are represented by nodes in the graph from the MRF_mod model. Colnames from this sample dataset must exactly match the colnames in the dataset that was used to fit the MRF_mod
MRF_mod	A fitted MRFcov or MRFcov_spatial model object

prep_covariates
: Logical flag stating whether to prep the dataset by cross-multiplication (TRUE by default; FALSE when used in other functions)

n_cores
: Positive integer stating the number of processing cores to split the job across. Default is 1 (no parallelisation)

progress_bar
: Logical. Progress bar in pbapply is used if TRUE, but this slows estimation.

Details

Observations for nodes in data are predicted using linear predictions from MRF_mod. If family = "binomial", a second element containing binary predictions for nodes is returned. Note that predicting values for unobserved locations using a spatial MRF is not currently supported

Value

A matrix containing predictions for each observation in data. If family = "binomial", a second element containing binary predictions for nodes is returned.

References

Clark, NJ, Wells, K and Lindberg, O. Unravelling changing interspecific interactions across environmental gradients using Markov random fields. (2018). Ecology doi: 10.1002/ecy.2221 Full text here.

See Also

MRFcov, cv_MRF_diag

Examples

```
data("Bird.parasites")
# Fit a model to a subset of the data (training set)
CRFmod <- MRFcov(data = Bird.parasites[1:300, ], n_nodes = 4, family = "binomial")

# If covariates are included, prep the dataset for gathering predictions
prepped_pred <- prep_MRF_covariates(Bird.parasites[301:nrow(Bird.parasites), ], n_nodes = 4)

# Predict occurrences for the remaining subset (test set)
predictions <- predict_MRF(data = prepped_pred, MRF_mod = CRFmod)

# Visualise predicted occurrences for nodes in the test set
predictions$Binary_predictions

# Predicting spatial MRFs requires the user to supply the spatially augmented dataset
data("Bird.parasites")
Latitude <- sample(seq(120, 140, length.out = 100), nrow(Bird.parasites), TRUE)
Longitude <- sample(seq(-19, -22, length.out = 100), nrow(Bird.parasites), TRUE)
coords <- data.frame(Latitude = Latitude, Longitude = Longitude)
CRFmod_spatial <- MRFcov_spatial(data = Bird.parasites, n_nodes = 4,
                                 family = 'binomial', coords = coords)
```

```
predictions <- predict_MRF(data = CRFmod_spatial$mrf_data,
                           prep_covariates  = FALSE,
                           MRF_mod = CRFmod_spatial)
```

`predict_MRFnetworks` *Extract predicted network metrics for observations in a given dataset using equations from a fitted* `MRFcov` *object*

Description

This function uses outputs from fitted `MRFcov` and `bootstrap_MRF` models to generate linear predictions for each observation in `data` and calculate probabilistic network metrics from weighted adjacency matrices.

Usage

```
predict_MRFnetworks(
  data,
  MRF_mod,
  cutoff,
  omit_zeros,
  metric,
  cached_predictions = NULL,
  prep_covariates,
  n_cores,
  progress_bar = FALSE
)
```

Arguments

`data`	Dataframe. The sample data where the left-most variables are variables that are represented by nodes in the graph. Colnames from this sample dataset must exactly match the colnames in the dataset that was used to fit the `MRF_mod`
`MRF_mod`	A fitted `MRFcov` or `bootstrap_MRF` object
`cutoff`	Single numeric value specifying the linear prediction threshold. Species whose linear prediction is below this level for a given observation in `data` will be considered absent, meaning they cannot participate in community networks. Default is `0.5` for `family == 'binomial'` or `0` for other families
`omit_zeros`	Logical. If `TRUE`, each species will not be considered to participate in community networks for observations in which that species was not observed in `data`. If `FALSE`, the species is still considered to have possibly occurred, based on the linear prediction for that observation. Default is `FALSE`
`metric`	The network metric to be calculated for each observation in `data`. Recognised values are : `"degree"`, `"eigencentrality"`, or `"betweenness"`, or leave blank to instead return a list of adjacency matrices

`cached_predictions`
: Use if providing stored predictions from `predict_MRF` to prevent unneccessary replication. Default is to calculate predictions first and then calculate network metrics

`prep_covariates`
: Logical flag stating whether to prep the dataset by cross-multiplication (`TRUE` by default; use `FALSE` for predicting networks from `MRFcov_spatial` objects)

`n_cores`
: Positive integer stating the number of processing cores to split the job across. Default is `1` (no parallelisation)

`progress_bar`
: Logical. Progress bar in pbapply is used if `TRUE`, but this slows estimation.

Details

Interaction parameters are predicted for each observation in `data` and then converted into a weighted, undirected adjacency matrix using `graph.adjacency`. Note that the network is probabilistic, as node occurrences/abundances are predicted using fitted model equations from `MRF_mod`. If a linear prediction for a given observation falls below the user-specified `cutoff`, the node is considered absent from the community and cannot participate in the network. After correcting for the linear predictions, the specified network metric (degree centrality, eigencentrality, or betweenness) for each observation in `data` is then calculated and returned in a `matrix`. If `metric` is not supplied, the weighted, undirected adjacency matrices are returned in a `list`

Value

Either a `matrix` with `nrow = nrow(data)`, containing each species' predicted network metric at each observation in `data`, or a `list` with `length = nrow(data)` containing the weighted, undirected adjacency matrix predicted at each observation in `data`

See Also

`MRFcov`, `bootstrap_MRF`, `degree`, `eigen_centrality`, `betweenness`

Examples

```
data("Bird.parasites")
CRFmod <- MRFcov(data = Bird.parasites, n_nodes = 4,
                 family = "binomial")
predict_MRFnetworks(data = Bird.parasites[1:200, ],
                    MRF_mod = CRFmod, metric = "degree",
                    cutoff = 0.25)
```

prep_MRF_covariates *Cross-multiply response and covariate variables*

Description

This function performs the cross-multiplication necessary for prepping datasets to be used in MRFcov models. This function is called by several of the functions within the package.

Usage

```
prep_MRF_covariates(data, n_nodes)
```

Arguments

data	Dataframe. The input data where the n_nodes left-most variables are outcome variables to be represented by nodes in the graph
n_nodes	Integer. The index of the last column in data which is represented by a node in the final graph. Columns with index greater than n_nodes are taken as covariates. Default is the number of columns in data, corresponding to no additional covariates

Details

Observations of nodes (species) in data are prepped for MRFcov analysis by multiplication. This function is not designed to be called directly, but is used by other functions in the package (namely MRFcov, MRFcov_spatial, cv_MRF_diag, and bootstrap_MRF)

Value

Dataframe of the prepped response and covariate variables necessary for input in MRFcov models

prep_MRF_covariates_spatial *Cross-multiply response and covariate variables and build spatial splines*

Description

This function performs the cross-multiplication necessary for prepping datasets to be used in MRFcov_spatial models.

Usage

```
prep_MRF_covariates_spatial(data, n_nodes, coords)
```

Arguments

data	Dataframe. The input data where the n_nodes left-most variables are outcome variables to be represented by nodes in the graph
n_nodes	Integer. The index of the last column in data which is represented by a node in the final graph. Columns with index greater than n_nodes are taken as covariates. Default is the number of columns in data, corresponding to no additional covariates
coords	A two-column dataframe (with nrow(coords) == nrow(data)) representing the spatial coordinates of each observation in data. Ideally, these coordinates will represent Latitude and Longitude GPS points for each observation. The coordinates are used to create smoothed Gaussian Process spatial regression splines via smooth.construct2. Here, the basis dimension of the smoothed term is chosen based on the number of unique GPS coordinates in coords. If this number is less than 100, then this number is used. If the number of unique coordiantes is more than 100, a value of 100 is used (this parameter needs to be large in order to ensure enough degrees of freedom for estimating 'wiggliness' of the smooth term; see choose.k for details).

Details

Observations of nodes (species) in data are prepped for MRFcov_spatial analysis by multiplication. This function is useful if users wish to prep the spatial splines beforehand and split the data manually for out-of-sample cross-validation. To do so, prep the splines here and set prep_splines = FALSE in MRFcov_spatial

Value

Dataframe of the prepped response and covariate variables necessary for input in MRFcov_spatial models

Index

∗ **datasets**
Bird.parasites, 2

betweenness, *18*
Bird.parasites, 2
bootstrap_MRF, 3, *10*, *13–15*, *17–19*

choose.k, *13*, *20*
createFolds, *7*
cv.glmnet, *4*, *7*, *8*, *10*, *11*
cv_MRF_diag, 5, *10*, *16*, *19*
cv_MRF_diag_rep, *7*
cv_MRF_diag_rep *(*cv_MRF_diag*)*, 5
cv_MRF_diag_rep_spatial *(*cv_MRF_diag*)*, 5

degree, *18*

eigen_centrality, *18*

graph.adjacency, *18*

makePSOCKcluster, *3*, *7*, *10*, *12*
MRFcov, *3–5*, *8*, 9, *12–19*
MRFcov_spatial, *4*, 12, *15*, *18–20*

plotMRF_hm, 14
predict_MRF, *4*, *8*, *10*, *13*, 15, *18*
predict_MRFnetworks, *13*, 17
prep_MRF_covariates, *10*, *12*, 19
prep_MRF_covariates_spatial, 19

scale, *3*, *6*
smooth.construct.gp.smooth.spec, *13*
smooth.construct2, *13*, *20*