Package ‘eicm’

July 22, 2025

Version 1.0.3

Date 2023-04-28

Title Explicit Interaction Community Models

Maintainer Miguel Porto <`mpbertolo@gmail.com`>

Depends R (>= 3.5.0)

Imports methods, parallel, GA (>= 3.1.1), snow, doSNOW, iterators, pso, ucminf, foreach, graphics, optimParallel

Suggests igraph, knitr, rmarkdown

VignetteBuilder knitr

Description Model fitting and species biotic interaction network topology selection for explicit interaction community models. Explicit interaction community models are an extension of binomial
linear models for joint modelling of species communities, that incorporate both the effects of species biotic interactions and the effects of missing covariates. Species interactions are modelled as direct effects of each species on each of the others, and are estimated alongside the effects of missing covariates, modelled as latent factors. The package includes a penalized maximum likelihood
fitting function, and a genetic algorithm for selecting the most parsimonious species interaction network topology.

License GPL (>= 2)

Encoding UTF-8

URL `https://github.com/miguel-porto/eicm`

BugReports `https://github.com/miguel-porto/eicm`

LazyData true

RoxygenNote 7.2.3

NeedsCompilation yes

Author Miguel Porto [aut, cre] (ORCID: <`https://orcid.org/0000-0001-7640-0623`>), Pedro Beja [aut] (ORCID: <`https://orcid.org/0000-0001-8164-0760`>)

Repository CRAN

Date/Publication 2023-05-05 23:10:02 UTC

Contents

`eicm-package` *eicm: Explicit Interaction Community Models*

Description

Explicit Interaction Community Models

Details

Model fitting and species biotic interaction network topology selection for explicit interaction community models. Explicit interaction community models are an extension of binomial linear models for joint modelling of species communities, that incorporate both the effects of species biotic interactions and the effects of missing covariates. Species interactions are modelled as direct effects of each species on each of the others, and are estimated alongside the effects of missing covariates, modelled as latent factors. The package includes a penalized maximum likelihood fitting function, and a genetic algorithm for selecting the most parsimonious species interaction network topology.

Main functions

`eicm`, `eicm.fit`

`as.eicm` *Define a parameterized EICM model*

Description

Constructs a EICM model object from given coefficients and data. Useful for simulating "true" models, otherwise only used internally.

Usage

```
as.eicm(
  env.coefs,
  sp.coefs = NULL,
  latent = NULL,
  options = NULL,
  occurrences = NULL,
  env = NULL,
  traits = NULL,
  intercept = TRUE,
  regularization = NULL
)
```

Arguments

`env.coefs`	the environmental coefficient matrix: a species x variable matrix (including intercept).
`sp.coefs`	the species interaction coefficient matrix: a species x species matrix, with zero diagonal.
`latent`	the values for the latent variables in each sample: a sample x latent variable matrix.
`options`	a `eicm.options` object defining options for fitting. Usually not needed, use `forbidden`, `mask.sp` and `exclude.prevalence` instead.
`occurrences`	a binary (0/1) sample x species matrix, possibly including NAs.
`env`	an optional sample x environmental variable matrix, for the known environmental predictors.
`traits`	an optional species x trait matrix. Currently, it is only used for excluding species interactions *a priori*.
`intercept`	logical specifying whether to add a column for the species-level intercepts.
`regularization`	a two-element numeric vector defining the regularization lambdas used for environmental coefficients and for species interactions respectively. See details.

Details

`regularization` is only used for storing the regularization lambdas used in model fitting. It is ignored in simulation.

Value

A `eicm` object that can be used for prediction.

Note

This function is only useful for simulation purposes. If you want to predict values from a fitted model, a `eicm` object is already provided for the fitted model.

See Also

`predict.eicm`

Examples

```
# Generate some coefficients
nenv <- 2
nsp <- 20
nsamples <- 200

env <- matrix(rnorm(nenv * nsamples), ncol=nenv, nrow=nsamples)
env.coefs <- matrix(runif((nenv + 1) * nsp, -4, 4), nrow=nsp)
sp.coefs <- matrix(0, nrow=nsp, ncol=nsp)
sp.coefs[3, 5] <- 3
sp.coefs[4, 8] <- 2

# Define a true model (including environmental data)
truemodel <- as.eicm(env=env, env.coefs=env.coefs, sp.coefs=sp.coefs)

# We can now realize it
predict(truemodel)
```

`coef.eicm` *Extract EICM model coefficients*

Description

Extract the EICM model coefficients, organized in three separate matrices.

Usage

```
## S3 method for class 'eicm'
coef(object, ...)
```

Arguments

`object`	a EICM model.
`...`	additional argument(s) for methods.

Value

A list with three coefficient matrices:

$env environmental coefficients

$sp species interaction coefficients. It reads as: species C (column) affects species R (row) with the coefficient sp[R, C]

$samples estimated latent variable values

`coefficientComparisonPlot`

Diagnostic monitor plots for EICM

Description

Visually compare the true model with estimation results (final results or during model fitting) and compute accuracy statistics.

Usage

```
coefficientComparisonPlot(
  model,
  true.model,
  nenv.to.plot = 0,
  nlatent.to.plot = 0,
  plot.interactions = any(model$sp != 0),
  plot.intercept = FALSE,
  excluded.interactions = NULL,
  layout = TRUE,
  noplot = FALSE,
  env.stats = TRUE,
  legend = TRUE,
  ...
)
```

Arguments

`model`	the EICM model of interest.
`true.model`	the true model to compare with (usually, the one used for simulating the data).
`nenv.to.plot`	the number of environmental variables to plot.
`nlatent.to.plot`	the number of latent variables to plot.
`plot.interactions`	logical. Plot interaction coefficient scatterplot?
`plot.intercept`	logical. plot the species-level intercepts?

`excluded.interactions`	a binary species x species matrix telling which interactions were excluded *a priori*.
`layout`	logical. Do multi-panel layout?
`noplot`	logical. Do plots? If TRUE, it will return the accuracy statistics only.
`env.stats`	logical. Compute accuracy for environmental predictors?
`legend`	logical. Plot legend?
`...`	further arguments to pass to `plot`.

Value

A vector with accuracy statistics.

`confint.eicm` *Confidence intervals for EICM parameters*

Description

Computes the profile (penalized) likelihood confidence intervals for all estimated parameters in a EICM model. If the likelihood profiles are not computed yet, they will be computed first.

Usage

```
## S3 method for class 'eicm'
confint(
  object,
  parm,
  level = 0.99,
  step = 0.3,
  ncores = parallel::detectCores(),
  ...
)
```

Arguments

`object`	the fitted EICM model.
`parm`	currently unused.
`level`	the confidence level required.
`step`	the step increments/decrements at which to compute the likelihood profile points.
`ncores`	the number of CPU cores to use when computing profiles for all parameters.
`...`	additional argument(s) for methods

Value

The same model object with a new `confint` component.

Examples

```
# load the included parameterized model
data(truemodel)

# realize the model
occurrences <- predict(truemodel, nrepetitions=1)

# fit the model without species interactions
fitted <- eicm(occurrences, n.latent=2, mask.sp=0, do.selection=FALSE)$fitted.model

# compute confidence intervals for all parameters
# this updates the fitted model with the confints
fitted <- confint(fitted, ncores=2)

# plot the confidence intervals
plot(fitted, type="confint")
```

---

eicm *Fit and select an Explicit Interaction Community Model (EICM)*

Description

Given species occurrence data and (optionally) measured environmental predictors, fits and selects an EICM that models species occurrence probability as a function of measured predictors, unmeasured predictors (latent variables) and direct species interactions.

Usage

```
eicm(
  occurrences,
  env = NULL,
  traits = NULL,
  intercept = TRUE,
  n.latent = 0,
  rotate.latents = FALSE,
  scale.latents = TRUE,
  forbidden = NULL,
  allowed = NULL,
  mask.sp = NULL,
  exclude.prevalence = 0,
  regularization = c(ifelse(n.latent > 0, 6, 0.5), 1),
  regularization.type = "hybrid",
  penalty = 4,
  theta.threshold = 0.5,
  latent.lambda = 1,
  fit.all.with.latents = TRUE,
  popsize.sel = 2,
```

```
  n.cores = parallel::detectCores(),
  parallel = FALSE,
  true.model = NULL,
  do.selection = TRUE,
  do.plots = TRUE,
  fast = FALSE,
  refit.selected = TRUE
)
```

Arguments

`occurrences`	a binary (0/1) sample x species matrix, possibly including NAs.
`env`	an optional sample x environmental variable matrix, for the known environmental predictors.
`traits`	an optional species x trait matrix. Currently, it is only used for excluding species interactions *a priori*.
`intercept`	logical specifying whether to add a column for the species-level intercepts.
`n.latent`	the number of latent variables to estimate.
`rotate.latents`	logical. Rotate the estimated latent variable values (the values of the latents at each sample) in the first step with PCA? Defaults to FALSE.
`scale.latents`	logical. Standardize the estimated latent variable values (the values of the latents at each sample) in the first step? Defaults to TRUE.
`forbidden`	a formula (or list of) defining which species interactions are not to be estimated. See details. This constraint is cumulative with other constraints (`mask.sp` and `exclude.prevalence`).
`allowed`	a formula (or list of) defining which species interactions are to be estimated. See details. This constraint is cumulative with other constraints (`mask.sp` and `exclude.prevalence`).
`mask.sp`	a scalar or a binary square species x species matrix defining which species interactions to exclude (0) or include (1) *a priori*. If a scalar (0 or 1), 0 excludes all interactions, 1 allows all interactions. If a matrix, species in the columns affect species in the rows, so, setting `mask.sp[3, 8] <- 0` means that species #8 is assumed *a priori* to not affect species #3. This constraint is cumulative with other constraints (`forbidden` and `exclude.prevalence`).
`exclude.prevalence`	exclude species interactions which are caused by species with prevalence equal or lower than this value. This constraint is cumulative with other constraints (`forbidden` and `mask.sp`)
`regularization`	a two-element numeric vector defining the regularization lambdas used for environmental coefficients and for species interactions respectively. See details.
`regularization.type`	one of "lasso", "ridge" or "hybrid", defining the type of penalty to apply. Type "hybrid" applies ridge penalty to environmental coefficients and LASSO to interaction coefficients.
`penalty`	the penalty applied to the number of species interactions to include, during variable selection.

`theta.threshold`	exclude species interactions (from network selection) whose preliminary coefficient (in absolute value) is lower than this value. This exclusion criterion is cumulative with the other user-defined exclusions.
`latent.lambda`	the regularization applied to latent variables and respective coefficients when estimating their values in samples.
`fit.all.with.latents`	logical. Whether to use the previously estimated latent variables when estimating the preliminary species interactions.
`popsize.sel`	the population size for the genetic algorithm, expressed as the factor to multiply by the recommended minimum. Ignored if `do.selection=FALSE`.
`n.cores`	the number of CPU cores to use in the variable selection stage and in the optimization.
`parallel`	logical. Whether to use `optimParallel` during optimizations instead of `optim`.
`true.model`	for validation purposes only: the true model that has generated the data, to which the estimated coefficients will be compared in each selection algorithm iteration.
`do.selection`	logical. Conduct the variable selection stage, over species interaction network topology?
`do.plots`	logical. Plot diagnostic and trace plots?
`fast`	a logical defining whether to do a fast - but less accurate - estimation, or a normal estimation.
`refit.selected`	logical. Refit with exact estimates the best model after network selection? Note that, for performance reasons, the models fit during the network selection stage use an approximate likelihood.

Details

An Explicit Interaction Community Model (EICM) is a simultaneous equation linear model in which each species model integrates all the other species as predictors, along with measured and latent variables.

This is the main function for fitting EICM models, and is preferred over using `eicm.fit` directly.

This function conducts the fitting and network topology selection workflow, which includes three stages: 1) estimate latent variable values; 2) make preliminary estimates for species interactions; 3) conduct network topology selection over a reduced model (based on the preliminary estimates).

The selection stage is optional. If not conducted, the species interactions are estimated (all or a subset according to the user-provided constraints), but not selected. See `vignette("eicm")` for commented examples on a priori excluding interactions.

Missing data in the response matrix is allowed.

Value

A `eicm.list` with the following components:

true.model: a copy of the `true.model` argument.

latents.only: the model with only the latent variables estimated.

fitted.model the model with only the species interactions estimated.

selected.model: the final model with all coefficients estimated, after network topology selection. This is the "best" model given the selection criterion (which depends on `regularization` and `penalty`.

When accessing the results, remember to pick the model you want (usually, `selected.model`). `plot` automatically picks `selected.model` or, if NULL, `fitted.model`.

See Also

`eicm-package`, `eicm.fit`, `plot.eicm`

Examples

```
# refer to the vignette for a more detailed explanation

# This can take some time to run

# Load the included parameterized model
data(truemodel)

# make one realization of the model
occurrences <- predict(truemodel, nrepetitions=1)

# Fit and select a model with 2 latent variables to be estimated and all
# interactions possible
m <- eicm(occurrences, n.latent=2, penalty=4, theta.threshold=0.5, n.cores=2)

plot(m)
```

`eicm.data` *Define a data object for a EICM model*

Description

Constructs a EICM data object for prediction. The data object contains all data matrices that may be needed for prediction. Usually, you don't need to invoke this function directly, use `as.eicm` instead.

Usage

```
eicm.data(occurrences = NULL, env = NULL, traits = NULL, intercept = TRUE)
```

Arguments

`occurrences`	a binary (0/1) sample x species matrix, possibly including NAs.
`env`	an optional sample x environmental variable matrix, for the known environmental predictors.
`traits`	an optional species x trait matrix. Currently, it is only used for excluding species interactions *a priori*.
`intercept`	logical specifying whether to add a column for the species-level intercepts.

Value

A `eicm.data` object that can be used for defining a model.

`eicm.fit` *Estimate a EICM model*

Description

Estimates the parameter values of a EICM model from the provided observation data. This is the low-level estimation function. Users should use `eicm` instead, particularly if estimating latent variables and species interactions.

Usage

```
eicm.fit(
  occurrences,
  env = NULL,
  traits = NULL,
  intercept = TRUE,
  n.latent = 0,
  forbidden = NULL,
  allowed = NULL,
  mask.sp = NULL,
  exclude.prevalence = 0,
  options = NULL,
  initial.values = NULL,
  regularization = c(ifelse(n.latent > 0, 0.5, 0), 1),
  regularization.type = "hybrid",
  fast = FALSE,
  n.cores = 1,
  optim.method = "L-BFGS-B",
 optim.control = list(trace = 1, maxit = 10000, ndeps = 1e-04, factr = ifelse(fast,
    1e-04, 1e-06)/.Machine$double.eps)
)
```

Arguments

`occurrences`	a binary (0/1) sample x species matrix, possibly including NAs.
`env`	an optional sample x environmental variable matrix, for the known environmental predictors.
`traits`	an optional species x trait matrix. Currently, it is only used for excluding species interactions *a priori*.
`intercept`	logical specifying whether to add a column for the species-level intercepts.
`n.latent`	the number of latent variables to estimate.
`forbidden`	a formula (or list of) defining which species interactions are not to be estimated. See details. This constraint is cumulative with other constraints (`mask.sp` and `exclude.prevalence`).
`allowed`	a formula (or list of) defining which species interactions are to be estimated. See details. This constraint is cumulative with other constraints (`mask.sp` and `exclude.prevalence`).
`mask.sp`	a scalar or a binary square species x species matrix defining which species interactions to exclude (0) or include (1) *a priori*. If a scalar (0 or 1), 0 excludes all interactions, 1 allows all interactions. If a matrix, species in the columns affect species in the rows, so, setting `mask.sp[3, 8] <- 0` means that species #8 is assumed *a priori* to not affect species #3. This constraint is cumulative with other constraints (`forbidden` and `exclude.prevalence`).
`exclude.prevalence`	exclude species interactions which are caused by species with prevalence equal or lower than this value. This constraint is cumulative with other constraints (`forbidden` and `mask.sp`)
`options`	a `eicm.options` object defining options for fitting. Usually not needed, use `forbidden`, `mask.sp` and `exclude.prevalence` instead.
`initial.values`	the starting values for all parameters. Used only for speeding up fitting when there are previous estimates available.
`regularization`	a two-element numeric vector defining the regularization lambdas used for environmental coefficients and for species interactions respectively. See details.
`regularization.type`	one of "lasso", "ridge" or "hybrid", defining the type of penalty to apply. Type "hybrid" applies ridge penalty to environmental coefficients and LASSO to interaction coefficients.
`fast`	a logical defining whether to do a fast - but less accurate - estimation, or a normal estimation.
`n.cores`	the number of CPU cores to use in the L-BFGS-B optimization. This may be reduced to prevent excessive memory usage.
`optim.method`	the optimization function to use. Should be set to the default.
`optim.control`	the optimization parameters to use. Should be set to the defaults.

Details

By default, all species interactions are estimated. Uers can control which species interactions are to be estimated with the arguments `forbidden`, `mask.sp` and `exclude.prevalence`, which place cumulative restrictions on which interactions to estimate. See `vignette("eicm")` for commented examples.

Value

A fitted `eicm` object.

Note

If estimating latent variables and species interactions, use `eicm` instead.

See Also

`eicm`, `confint.eicm`, `plot.eicm`

Examples

```
# Simulate some random occurrence data
nenv <- 2
nsp <- 10
nsamples <- 200

env <- matrix(rnorm(nenv * nsamples), ncol=nenv, nrow=nsamples)
env.coefs <- matrix(runif((nenv + 1) * nsp, -4, 4), nrow=nsp)
sp.coefs <- matrix(0, nrow=nsp, ncol=nsp)
sp.coefs[3, 5] <- 3
sp.coefs[4, 8] <- 2

# Define a true model
truemodel <- as.eicm(env=env, env.coefs=env.coefs, sp.coefs=sp.coefs)

# realize the model
simulated.data <- predict(truemodel, nrepetitions=1)


# fit the model without species interactions
fittedNoInt <- eicm.fit(simulated.data, env, mask.sp=0)

# fit the model with all species interactions
fittedInt <- eicm.fit(simulated.data, env, mask.sp=1)

# compute confidence intervals for all parameters
fittedInt <- confint(fittedInt, ncores=2)

# plot estimated parameters and confidence intervals
plot(fittedInt, type="confint", truemodel=truemodel)
```

`eicm.matrix` *Define a model object for a EICM model*

Description

Constructs a EICM model object for prediction. The model object contains all coefficient matrices that may be needed for prediction. Usually, you don't need to invoke this function directly, use `as.eicm` instead.

Usage

```
eicm.matrix(env.coefs, sp.coefs = NULL, latent = NULL, options = NULL)
```

Arguments

`env.coefs`	the environmental coefficient matrix: a species x variable matrix (including intercept).
`sp.coefs`	the species interaction coefficient matrix: a species x species matrix, with zero diagonal.
`latent`	the values for the latent variables in each sample: a sample x latent variable matrix.
`options`	options for the model fitting.

Details

The EICM model is a list composed of three matrices plus the fitting options:

1. env
2. sp
3. samples

Value

A `eicm.matrix` object that can be used for defining a model.

`eicm.options` *Set EICM fitting options*

Description

Construct a EICM options object to inform the fitting engine. There is usually no need to use this function directly.

Usage

```
eicm.options(...)
```

Arguments

`...`	a named list of options. See details.

Details

Possible options are currently: `mask` and `offset`. Both are lists having the same structure of an `eicm.matrix` object:

- **$mask** Binary matrices defining which coefficients are to be estimated in model fitting OR a scalar, constant for all coefficients. 0 or FALSE exclude the given term from estimation, i.e., fixes it at 0.
  - **$mask$env** environmental coefficient mask
  - **$mask$sp** species interaction mask
- **$offset** Numeric matrices defining constant terms, to be fixed and not estimated
  - **$offset$env** environmental coefficient offset
  - **$offset$sp** species interaction offset

When an offset for a term is nonzero, the respective mask element will be automatically zeroed (so it is not estimated).

Value

A `eicm.options` object with options for model fitting, currently a model mask and model offsets.

`generateEICM` *Generate EICM model following a Beta frequency distribution*

Description

Generates a randomly parameterized EICM model (predictors, environmental coefficients and species interactions), ensuring that communities simulated with this model have a frequency distribution that matches the given Beta distribution of frequencies as much as possible.

Usage

```
generateEICM(
  nspecies,
  nsamples,
  nenv,
  ninteractions,
  shape1,
  shape2,
  nbins = 10,
  nrepetitions = 5,
  shrinkage = 2,
  bounds = 10,
  swarm.size = floor((ninteractions + nspecies * (nenv + 1)) * 0.5),
  maxit.stagnate = 150
)
```

Arguments

`nspecies`	the number of species to generate.
`nsamples`	the number of samples to generate.
`nenv`	the number of environmental predictors to generate.
`ninteractions`	the number of species interactions to generate.
`shape1`	the shape1 parameter of the Beta distribution.
`shape2`	the shape2 parameter of the Beta distribution.
`nbins`	the number of histogram bins for comparing distributions.
`nrepetitions`	the number of times to realize a model candidate to average their distributions.
`shrinkage`	the shrinkage factor for generated coefficients, when computing fitness criterion. Ensures that the generated coefficients remain at plausible values.
`bounds`	the allowed range for the coefficients `c(-bounds, +bounds)`.
`swarm.size`	the swarm size of the particle swarm optimization.
`maxit.stagnate`	the number of iterations without improvement until the optimization stops.

Details

This function is useful for generating a realistic random model for simulation, i.e. a model that, when simulated, will yield species communities with a distribution of frequencies akin of real communities: with most species being rare. The generated coefficients are not assumed to follow any distribution, but are optionally shrinked so that their values will remain "decent". The values of the environmental predictors are drawn from a gaussian distribution.

Value

A EICM model of class `eicm`

Examples

```
# Generate model with 32 species, 30 species interactions and 2 environmental predictors
# for 500 samples with a frequency distribution following a Beta(1.5, 3)
model <- generateEICM(nspecies=32, nsamples=500, nenv=2, ninteractions=30,
  shape1=1.5, shape2=3)

# make one realization
data <- predict(model, nrepetitions=1)

# plot frequency histogram: should follow a Beta distribution.
hist(apply(data, 2, sum) / nrow(data), breaks=seq(0, 1, by=0.1), xlim=c(0, 1),
  main="Frequency distribution of one realization", xlab="Frequency in samples",
  ylab="Number of species")
```

`interactionPlot` *Compare two interaction matrices*

Description

Visually compare two interaction (adjacency) matrices and return accuracy statistics

Usage

```
interactionPlot(
  estimated.adjacency.matrix,
  true.adjacency.matrix,
  excluded.interactions = NULL,
  labels = TRUE,
  noplot = FALSE,
  legend = TRUE,
  ...
)
```

Arguments

`estimated.adjacency.matrix`	the interaction matrix of the model of interest.
`true.adjacency.matrix`	the interaction matrix of the model to compare with (usually, the one used for simulating the data).
`excluded.interactions`	a binary species x species matrix telling which interactions were excluded *a priori*.
`labels`	logical. Add default axis labels and title?
`noplot`	logical. Do plots? If TRUE, it will return the accuracy statistics only.
`legend`	logical. Plot legend?
`...`	further arguments to pass to `plot`.

Value

A vector with accuracy statistics.

`isCyclic` *Is the adjacency matrix a cyclic graph?*

Description

Tests whether the given graph (given as an adjacency matrix) contains cycles.

Usage

```
isCyclic(coefs)
```

Arguments

`coefs`	a square adjacency matrix.

Value

A logical indicating whether there are cycles in the graph.

`logLik.eicm` *EICM (penalized) log-likelihood*

Description

Compute the (penalized) log-likelihood of the data matrix included in the EICM model, or the log-likelihood of a new occurrence data matrix given the model.

Usage

```
## S3 method for class 'eicm'
logLik(object, occurrences = NULL, allow.na = TRUE, ...)
```

Arguments

`object`	a EICM model
`occurrences`	the occurrence data matrix. If omitted, the data matrix used to fit the model is used.
`allow.na`	logical. Allow NAs in the occurrence matrix? If no NAs exist, it's faster to set to FALSE.
`...`	additional argument(s) for methods.

Value

A `logLik` object.

`plot.confint.eicm` *Plot EICM estimates and confidence intervals*

Description

Plot EICM estimates and confidence intervals, in a dot-and-whisker plot.

Usage

```
## S3 method for class 'confint.eicm'
plot(x, truemodel = NULL, ...)
```

Arguments

`x`	a eicm.confint object
`truemodel`	for validation purposes only. The true model used to simulate data.
`...`	other arguments passed to other functions.

Value

NULL.

`plot.eicm` *Quick plot EICM results*

Description

Multiple types of plots for EICM models: coefficients, network topology, confidence intervals and likelihood profiles. Allows to plot a single model or the comparison between an estimated and a true model.

Usage

```
## S3 method for class 'eicm'
plot(
  x,
  type = ifelse(is.null(true.model), "network", "coefficients"),
  true.model = NULL,
  ...
)
```

Arguments

`x`	a EICM model.
`type`	character. The type of plot, one of `confint`, `profile`, `network` or `coefficients`. See details.
`true.model`	the true model to compare with (usually, the one used for simulating the data).
`...`	further arguments to pass to `coefficientComparisonPlot` or other plotting functions.

Details

If no `true.model` is provided, `type` must be one of `confint`, `profile`, `network`.

If `true.model` is provided, `type` must be one of `network` or `coefficients`. In the latter case, see `coefficientComparisonPlot` for possible options.

If `x` is of type `eicm.list` (as returned by `eicm`), this function first tries to plot the model after network selection, then, if it was not computed, the fitted model with the full network.

Value

If `true.model` is provided and `type="coefficients"`, returns a named vector with accuracy statistics (invisibly). Otherwise, NULL.

See Also

`coefficientComparisonPlot`, `confint.eicm`

`plot.eicm.list` *Quick plot EICM results*

Description

Multiple types of plots for EICM models: coefficients, network topology, confidence intervals and likelihood profiles. Allows to plot a single model or the comparison between an estimated and a true model.

Usage

```
## S3 method for class 'eicm.list'
plot(
  x,
  type = ifelse(is.null(true.model), "network", "coefficients"),
  true.model = NULL,
  ...
)
```

Arguments

`x`	a EICM model.
`type`	character. The type of plot, one of `confint`, `profile`, `network` or `coefficients`. See details.
`true.model`	the true model to compare with (usually, the one used for simulating the data).
`...`	further arguments to pass to `coefficientComparisonPlot` or other plotting functions.

Details

If no `true.model` is provided, `type` must be one of `confint`, `profile`, `network`.

If `true.model` is provided, `type` must be one of `network` or `coefficients`. In the latter case, see `coefficientComparisonPlot` for possible options.

If `x` is of type `eicm.list` (as returned by `eicm`), this function first tries to plot the model after network selection, then, if it was not computed, the fitted model with the full network.

Value

If `true.model` is provided and `type="coefficients"`, returns a named vector with accuracy statistics (invisibly). Otherwise, NULL.

See Also

`coefficientComparisonPlot`, `confint.eicm`

`plot.profile.eicm` *Plot EICM likelihood profile*

Description

Plot one likelihood profile with the line representing a confidence level threshold.

Usage

```
## S3 method for class 'profile.eicm'
plot(x, level = 0.99, ...)
```

Arguments

`x`	a profile.eicm object, created with `profile.eicm`.
`level`	the significance level desired.
`...`	other arguments passed to other functions.

Value

NULL.

`plotNetworkFromMatrix` *Plot graphs from adjacency matrices*

Description

Plots a graph from a weighted adjacency matrix, using `igraph`'s plotting functions, optionally comparing it with another "true" adjacency matrix.

Usage

```
plotNetworkFromMatrix(
  adjacency,
  true.adjacency = NULL,
  labels = TRUE,
  exclude.orphans = TRUE,
  lwd = 1,
  edge.arrow.size = 0.8,
  severe.threshold = 0.5
)
```

Arguments

`adjacency`	the square adjacency matrix.
`true.adjacency`	optional. The reference "true" square adjacency matrix to which to compare the first one.
`labels`	logical. Draw default labels?
`exclude.orphans`	logical. Hide nodes without links?
`lwd`	a numerical value giving the amount by which the arrows' line width should be magnified relative to the default, when plotting the weighted graph (only used when `true.adjacency` is not provided).
`edge.arrow.size`	size of the arrow heads. See `igraph::plot.igraph`
`severe.threshold`	the absolute threshold above which the interaction weights are highlighted in the graph.

Details

When comparing two adjacency matrices

Value

The corresponding igraph network, invisibly.

Note

The arrow direction depicts the direction of the interaction. Species in columns affect species in rows.

The matrices should include row and column labels, otherwise the node labels may not correspond to the species index (when `exclude.orphans = TRUE`)

Examples

```
# generate two adjacency matrices with 15 species and 10 interactions
A <- matrix(0, ncol=15, nrow=15)
A[sample(length(A), 10)] <- runif(10)

B <- matrix(0, ncol=15, nrow=15)
B[sample(length(B), 10)] <- runif(10)

# set the species names
rownames(A) <- rownames(B) <-
  colnames(A) <- colnames(B) <- paste0("S", 1:15)

plotNetworkFromMatrix(A, B)
```

`predict.eicm` *Predict method for EICM fits*

Description

Obtains probability predictions (conditional or unconditional) or a model realization from a parameterized EICM model.

Usage

```
## S3 method for class 'eicm'
predict(object, nrepetitions = 10000, newdata = NULL, ...)
```

Arguments

`object`	the fitted EICM model.
`nrepetitions`	the number of realizations to conduct for computing probabilities. Set to 1 if you only need simulated community data.
`newdata`	optionally, a matrix in which to look for variables with which to predict. If omitted, the original data (used to fit the model) is used.
`...`	not used.

Details

The interaction network of the model must be an **acyclic graph**. Predictions are obtained by realizing the model multiple times and averaging realizations, because there is not a closed-form expression for their calculation.

To obtain conditional predictions, include presence/absence columns with species names in `newdata`. Named columns for all the environmental predictors must always be included.

Value

A species x sample matrix with predictions. If nrepetitions=1, predictions correspond to one realization, otherwise they are probabilities.

Note

If the eicm was fit without regularization, unconditional predictions are numerically equal to those of simple binomial GLMs.

Examples

```
# Load the included parameterized model
data(truemodel)

# for reference, plot the species interaction network
plot(truemodel, type="network")
```

```
# Unconditional predictions
# let's fix environmental predictors at 0, for simplicity.
predict(truemodel, newdata=cbind(env01=0, env02=0))

# Conditional predictions
# predict probabilities for all species conditional on the
# known presence of sp011 (compare sp014 and sp004 with the above)
predict(truemodel, newdata=cbind(env01=0, env02=0, sp011=1))

# Propagation of indirect species effects
# predict probabilities for all species conditional on the known
# absence (first line) and known presence (second line) of sp005 and sp023
predict(truemodel, newdata=cbind(env01=0, env02=0, sp012=c(0, 1), sp018=c(0, 1)), nrep=100000)

# Notice the effects on sp026 and the effect propagation to those
# species which depend on it (sp013, sp008)
# Also compare with unconditional predictions
```

`print.eicm.matrix` *Print EICM model matrix*

Description

Prints an excerpt of the EICM model coefficients.

Usage

```
## S3 method for class 'eicm.matrix'
print(x, ...)
```

Arguments

`x`	a EICM model matrix.
`...`	additional argument(s) for methods.

Value

NULL.

`profile.eicm` *Likelihood profiles for EICMs*

Description

Computes the profile (penalized) likelihood for all (or only one) estimated parameters in a EICM model.

Usage

```
## S3 method for class 'eicm'
profile(
  fitted,
  all.pars = TRUE,
  parmatrix,
  species,
  parameter,
  step = 0.3,
  ncores = parallel::detectCores(),
  alpha = 0.01,
  ...
)
```

Arguments

`fitted`	the fitted EICM model.
`all.pars`	logical. Compute for all model parameters?
`parmatrix`	if all.pars=FALSE, in which matrix is the parameter of interest, "env" or "sp"?
`species`	if all.pars=FALSE, in which row of `parmatrix` is the parameter of interest?
`parameter`	if all.pars=FALSE, in which column of `parmatrix` is the parameter of interest?
`step`	the step increments/decrements at which to compute the likelihood profile points.
`ncores`	the number of CPU cores to use when computing profiles for all parameters.
`alpha`	highest significance level that will be guaranteed for this profile.
`...`	additional argument(s) for methods

Details

Likelihod profiles will use the same regularization settings that were used in model fitting.

Value

The same model object updated with a new `profile` component.

Note

Confidence intervals should **not** be computed on a model whose terms have been selected.

This function is optimized for computing profiles of multiple parameters simultaneously (in parallel).

Examples

```
# load the included parameterized model
data(truemodel)

# realize the model
occurrences <- predict(truemodel, nrepetitions=1)

# fit the model without species interactions
fitted <- eicm(occurrences, n.latent=2, mask.sp=0, do.selection=FALSE)$fitted.model

# compute likelihood profiles for all parameters
fitted <- profile(fitted, ncores=2)

# plot the first 9 profiles
par(mfrow=c(3, 3))
dummy <- lapply(fitted$profile[1:9], plot)
```

`truemodel` *A parameterized EICM model for simulation*

Description

A parameterized EICM model ready for simulating communities with a frequency distribution following a Beta distribution with shape1=1.5 and shape2=3.

Usage

```
truemodel
```

Format

A 'eicm' object with 2 environmental predictors in 400 samples, 32 species and 30 species interactions.

Details

The model was generated with the command:

`truemodel <- generateEICM(32, 400, 2, 30, shape1=1.5, shape2=3)`

Index