Package: BayesianNetworks (via r-universe)

August 15, 2024

Title Bayesian Modelling of Bipartite Networks

Version 0.0.7

Description Bayesian modelling of bipartite network structure, following the approach of Young et al. <doi:10.1038/s41467-021-24149-x>.

License GPL (>= 3)

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

URL `https://github.com/Pakillo/BayesianNetworks`

BugReports `https://github.com/Pakillo/BayesianNetworks/issues`

Imports cmdstanr, dplyr, ggplot2, network.tools, tidybayes

Remotes stan-dev/cmdstanr, Pakillo/network.tools

Depends R (>= 4.1)

LazyData true

Repository https://pakillo.r-universe.dev

RemoteUrl https://github.com/Pakillo/BayesianNetworks

RemoteRef HEAD

RemoteSha 8436b0b165a42e5daadd4aa7e64956a3910d51bd

Contents

`check_model` *Check fitted model*

Description

Check fitted model

Usage

```
check_model(fit = NULL, data = NULL)
```

Arguments

`fit`	A fitted model, as obtained from `fit_model()`.
`data`	Data list (from `prepare_data()`).

Value

Model checks on console and graphical window.

Examples

```
data(web)
dt <- prepare_data(mat = web, sampl.eff = rep(20, nrow(web)))
fit <- fit_model(dt, refresh = 0)
check_model(fit, data = data)
```

`fit_model` *Fit model*

Description

Fit model

Usage

```
fit_model(
  data = NULL,
  model = c("sampling_effort", "Young2021", "varying_preferences"),
  beta = 0.01,
  ...
)
```

Arguments

`data` A named list containing the required data, as obtained from `prepare_data()`.

`model` character. One of "Young2021", "sampling_effort", or "varying_preferences", or a path to a file describing the Stan model in case you want to use a modified Stan model.

`beta` Rate of exponential prior on `r` (preference) parameter. Default beta is 0.01. Increase it if you have large count numbers (can examine the resultant prior using `plot_prior()`).

`...` Further arguments for `cmdstanr::sample()`, like `iter_warmup`, `iter_sampling`, or `thin`, among others. It is recommended to increase the number of iterations (e.g. iter_sampling = 10000).

Value

A fitted model (`cmdstanr::CmdStanMCMC()` object).

Examples

```
data(web)
dt <- prepare_data(mat = web, sampl.eff = rep(20, nrow(web)))
fit <- fit_model(dt)
```

`get_posterior` *Get posterior values*

Description

Get posterior values

Usage

```
get_posterior(
  fit = NULL,
  data = NULL,
  param = c("all", "connectance", "preference", "plant.abund", "animal.abund",
    "int.prob", "link")
)
```

Arguments

`fit`	Fitted model (from `fit_model()`)
`data`	Data list (from `prepare_data()`)
`param`	character. Name of the parameter to retrieve the posterior samples.

Value

A data frame

Examples

```
data(web)
dt <- prepare_data(mat = web, sampl.eff = rep(20, nrow(web)))
fit <- fit_model(dt, refresh = 0)
get_posterior(fit, dt, param = "connectance")

int.prob <- get_posterior(fit, dt, param = "int.prob")
int.prob
int.prob |> tidybayes::mean_qi()  # mean edge probability

# all posteriors
get_posterior(fit, dt, param = "all")
```

`get_seed` *Get seed used to fit a model*

Description

Get seed used to fit a model

Usage

```
get_seed(fit = NULL)
```

Arguments

`fit`	A fitted model

Value

A number

Examples

```
data(web)
dt <- prepare_data(mat = web, sampl.eff = rep(20, nrow(web)))
fit <- fit_model(dt, refresh = 0)
get_seed(fit)
```

plot_counts_obs *Plot heatmap of observed counts*

Description

Plot heatmap of observed counts

Usage

```
plot_counts_obs(mat = NULL, ...)
```

Arguments

mat	A matrix with count data reporting interaction frequency (e.g. visits to flowers, number of fruits consumed per plant or species). Plants must be in rows, Animals must be in columns.
...	Further arguments for network.tools::plot_web_heatmap().

Value

A ggplot object

Examples

```
data(web)
plot_counts_obs(web)
plot_counts_obs(web, sort = FALSE)
plot_counts_obs(web, zero.na = FALSE, sort = FALSE)
```

plot_counts_pred *Plot heatmap of predicted counts*

Description

Plot heatmap of predicted counts

Usage

```
plot_counts_pred(pred.df = NULL, ...)
```

Arguments

pred.df	A data frame containing the predicted counts, as generated by predict_counts()
...	Further arguments for network.tools::plot_web_heatmap().

Value

A ggplot object

Examples

```
data(web)
dt <- prepare_data(mat = web, sampl.eff = rep(20, nrow(web)))
fit <- fit_model(dt, refresh = 0)
pred.df <- predict_counts(fit, dt)
plot_counts_pred(pred.df)
plot_counts_pred(pred.df, sort = FALSE)
```

plot_counts_pred_obs *Plot predicted versus observed counts*

Description

Plot predicted versus observed counts

Usage

```
plot_counts_pred_obs(
  pred.df = NULL,
  data = NULL,
  byplant = FALSE,
  width = 0.95,
  ...
)
```

Arguments

pred.df	A data frame containing the predicted counts, as generated by `predict_counts()`
data	Data list (from `prepare_data()`)
byplant	Logical. If TRUE, show predicted and observed counts per plant (using `ggplot2::facet_wrap()`). If FALSE, show all interactions in the same plot.
width	width of the credible interval (default = 0.95).
...	Further arguments to be passed to `ggplot2::facet_wrap()` if `byplant = TRUE`, or to `tidybayes::geom_pointinterval()` if `byplant = FALSE`.

Value

A ggplot object

Examples

```
data(web)
dt <- prepare_data(mat = web, sampl.eff = rep(20, nrow(web)))
fit <- fit_model(dt, refresh = 0)
pred.df <- predict_counts(fit, dt)
plot_counts_pred_obs(pred.df, dt)
plot_counts_pred_obs(pred.df, dt, fatten_point = 3)
plot_counts_pred_obs(pred.df, dt, byplant = TRUE)
plot_counts_pred_obs(pred.df, dt, byplant = TRUE, scale = "free")
```

`plot_interaction_prob` *Plot heatmap of interaction probabilities*

Description

Plot a heatmap of average interaction probabilities

Usage

```
plot_interaction_prob(post = NULL)
```

Arguments

`post`	Data frame containing the posterior probabilities, as generated from `get_posterior()`.

Value

A ggplot object

Examples

```
data(web)
dt <- prepare_data(mat = web, sampl.eff = rep(20, nrow(web)))
fit <- fit_model(dt, refresh = 0)
post <- get_posterior(fit, dt)
plot_interaction_prob(post)
```

plot_prior *Plot prior distribution for r (preference) parameter*

Description

The r (preference) parameter in Young et al. model takes a prior exponential distribution with rate = beta. Use this function to visualise the prior distribution of r given the chosen beta. Alternatively, if providing the fitted model, a plot comparing the prior versus posterior preference(s) will be produced.

Usage

```
plot_prior(beta = NULL, fit = NULL, data = NULL)
```

Arguments

beta	A number > 0. Rate of the exponential distribution.
fit	A fitted model, as obtained from fit_model().
data	Data list (from prepare_data()).

Value

A plot

Examples

```
## Providing value for beta
plot_prior(beta = 0.01)
plot_prior(beta = 0.001)

## Providing fitted model
data(web)
dt <- prepare_data(mat = web, sampl.eff = rep(20, nrow(web)))
fit <- fit_model(dt, refresh = 0)
plot_prior(fit = fit, data = dt)
```

plot_residuals *Plot heatmap of residuals*

Description

Plot heatmap of residuals (observed - predicted counts).

Usage

```
plot_residuals(pred.df = NULL, data = NULL, ...)
```

Arguments

`pred.df` A data frame containing the predicted counts, as generated by `predict_counts()`

`data` Data list (from `prepare_data()`)

`...` Further arguments for `network.tools::plot_web_heatmap()`.

Value

A ggplot object

Examples

```
data(web)
dt <- prepare_data(mat = web, sampl.eff = rep(20, nrow(web)))
fit <- fit_model(dt, refresh = 0)
pred.df <- predict_counts(fit, dt)
plot_residuals(pred.df, dt)
plot_residuals(pred.df, dt, sort = FALSE)
```

`predict_counts` *Predict interaction counts*

Description

Generate the posterior predictive distribution of counts for every pairwise interaction.

Usage

```
predict_counts(fit = NULL, data = NULL)
```

Arguments

`fit` Fitted model

`data` Data list

Value

A data frame

Examples

```
data(web)
dt <- prepare_data(mat = web, sampl.eff = rep(20, nrow(web)))
fit <- fit_model(dt, refresh = 0)
predict_counts(fit, dt)
```

prepare_data *Prepare the data for modelling*

Description

Prepare the data for modelling

Usage

```
prepare_data(mat = NULL, sampl.eff = NULL)
```

Arguments

`mat`	An integer matrix containing quantitative (not qualitative or binary) count data con interaction frequency (e.g. visits to flowers, number of fruits consumed per plant or species). Plants must be in rows, Animals must be in columns.
`sampl.eff`	A numeric vector with the sampling effort (e.g. observation hours) spent on each plant.

Value

A named list with all the data required to run the model.

Examples

```
data(web)
prepare_data(web, sampl.eff = rep(20, nrow(web)))
```

web *Plant-pollinator network*

Description

An example bipartite network of 8 plant species and 21 pollinators, from Kaiser-Bunbury et al. 2017.

Usage

```
web
```

Format

`web`**:**

A numeric (integer) matrix with 8 rows (representing plants) and 21 columns (representing animals)

Source

Kaiser-Bunbury, C., Mougal, J., Whittington, A. et al. Ecosystem restoration strengthens pollination network resilience and function. Nature 542, 223–227 (2017). https://doi.org/10.1038/nature21071

Index