Package ‘wispack’

June 8, 2025

Type Package

Encoding UTF-8

Title Implements warped-sigmoid Poisson-process mixed-effects models (WSPmm)

Version 1.0

Date 2025-03-17

Author Michael Barkasi

Maintainer Michael Barkasi <barkasi@wustl.edu>

Description WSPmm are intended for hypothesis testing of spatially or temporally variable count data. They parameterize data in terms of blocks separated by transition points, each block having a process rate and each transition point having both a position and slope.

License GPL (>= 3)

Imports Rcpp (>= 1.0.13-1), RcppEigen, methods, dtwclust

LinkingTo Rcpp, RcppEigen, StanHeaders, BH

Depends ggplot2, grid, gridExtra

SystemRequirements NLopt

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

NeedsCompilation yes

Contents

analyze.residuals *Analyze residuals from wisp fit*

Description

This function takes a vector mu.B of sampled values of a variable X, and a single observation mu.obs of the same variable, and computes the p-value of mu.obs using the empirical cumulative distribution function (ecdf) of mu.B.

Usage

```
analyze.residuals(wisp.results, verbose = TRUE)
```

Arguments

wisp.results	List, output of the wisp function.
verbose	Logical, if TRUE, prints information during the statistical analysis.

Value

Numeric p-value.

demo.sigmoid.plots *Make plot demonstrating how the wisp function is determined by the Rt, slope, and tpoint parameters*

Description

This function takes user-provided values for wisp function parameters and produces a panel of plots showing how the wisp model is determined by the Rt, slope, and tpoint parameters.

Usage

```
demo.sigmoid.plots(
  r = 4,
  s = 1,
  Rt = c(6, 3, 0.2, 6) * 4.65,
  tslope = c(0.4, 0.75, 1),
  tpoint = c(15, 38, 80)
)
```

Arguments

r	Numeric, upper asymptote for logistic (must be a single value).
s	Numeric, slope scalar at inflection point (must be a single value).
Rt	Numeric vector, rate parameters for the wisp function. Degree of the wisp model will be length of this vector minus 1.
tslope	Numeric vector, slope scalars for the wisp function. Must be one less than the length of Rt.
tpoint	Numeric vector, transition points for the wisp function. Must be one less than the length of Rt.

Value

Nothing. A ggplot object is printed to console.

demo.warp.plots *Make plot demonstrating how the wisp function is warped by the warping factors*

Description

This function takes user-provided values for wisp function parameters and produces a panel of plots showing how the wisp model is warped by the warping factors

Usage

```
demo.warp.plots(
  w = 2,
  point_pos = 60,
  point_neg = 40,
  Rt = c(6, 3, 0.2, 6) * 4.65,
  tslope = c(0.4, 0.75, 1),
  tpoint = c(15, 38, 80),
  w_factors = c(0.6, -0.9, 0.5)
)
```

Arguments

w	Numeric, warping factor (must be a single value).
point_pos	Numeric, x coordinate at which to place positive warp segment (must be a single value).
point_neg	Numeric, x coordinate at which to place negative warp segment (must be a single value).
Rt	Numeric vector, rate parameters for the wisp function. Degree of the wisp model will be length of this vector minus 1.
tslope	Numeric vector, slope scalars for the wisp function. Must be one less than the length of Rt.
tpoint	Numeric vector, transition points for the wisp function. Must be one less than the length of Rt.

`w_factors`	Numeric vector, warping factors for the wisp function. Must be length 3. First element is the warping factor for Rt, second for tslope, and third for tpoint. Note that this is more restrictive than the real wisp model, which not only allows for different warping factors across the model components (Rt, tslope, and tpoint), but also across the different elements within each model component as well.

Value

Nothing. A ggplot object is printed to console.

plot.child.summary *Print rate-count, residual, and parameter plots for one child level together.*

Description

Function to summarize all important information for an individual child level on one plot.

Usage

```
## S3 method for class 'child.summary'
plot(wisp.results, these.parents = NULL, these.childs = NULL, verbose = TRUE)
```

Arguments

`wisp.results`	List, output of the wisp function.
`these.parents`	Character vector, optional, specifies which parent levels to summarize. Defaults to all.
`these.childs`	Character vector, optional, specifies which child levels to summarize. Defaults to all.
`verbose`	Logical, if TRUE, prints information during plotting.

Value

Nothing, plots are printed to the current graphics device.

plot.decomposition *Plot individual components of wisp fit for a single child level*

Description

Extension of `plot.ratecount` which plots the individual pieces of the rate-count plot for a single child separately. Returns individual plots for data points only, fit lines only, and data points plus fit lines for individual random levels.

Usage

```
## S3 method for class 'decomposition'
plot(wisp.results, child, log = FALSE, dim.boundaries = c(), y.lim = NULL)
```

Arguments

`wisp.results`	List, output of the wisp function.
`child`	Character string, the child level to plot. Must be provided, and only one at a time.
`log`	Logical, if TRUE, plots on a log scale. Defaults to FALSE.
`dim.boundaries`	Numeric vector, independent block boundaries to plot for comparison. If empty, the argument is ignored.
`y.lim`	Numeric vector of length 2, limits for the y-axis of the plots. If NA, defaults to automatic limits.

Value

List of ggplot objects containing the decomposed rate-count plots.

`plot.effect.dist` *Plot parameter distributions from WISP results as histograms*

Description

Function to make nicely formatted histograms of fitted parameters from WISP results.

Usage

```
## S3 method for class 'effect.dist'
plot(wisp.results, verbose = TRUE)
```

Arguments

`wisp.results`	List, output of the wisp function.
`verbose`	Logical, if TRUE, prints information during plotting.

Value

List of ggplot objects containing histograms of fitted parameters.

`plot.MCMC.bs.comparison` *Visually compare normality and autocorrelation of bootstrap and MCMC parameter estimates*

Description

Function allowing for visual comparison of the parameter estimates from bootstrapping vs MCMC simulation. Shows density for ten representative parameters from bootstrapping and MCMC walk, the distributions of Shapiro-Walk p-values for bootstrapping vs MCMC walks, and the density of autocorrelation results for bootstrapping vs MCMC walks.

Usage

```
## S3 method for class 'MCMC.bs.comparison'
plot(wisp.results, verbose = TRUE)
```

Arguments

`wisp.results`	List, output of the wisp function.
`verbose`	Logical, if TRUE, prints information during plotting.

Value

List of ggplot objects containing plots of representative parameter distributions, Shapiro-Wilk normality test results, and autocorrelation plots for bootstrap and MCMC parameter estimates.

`plot.MCMC.walks` *Plot sampling of random walks and negative log likelihood from MCMC simulations*

Description

Function to make nicely formatted histograms of fitted parameters from WISP results.

Usage

```
## S3 method for class 'MCMC.walks'
plot(wisp.results, verbose = TRUE, low_samples = 10)
```

Arguments

`wisp.results`	List, output of the wisp function.
`verbose`	Logical, if TRUE, prints information during plotting.
`low_samples`	Integer, number of low-value parameters to plot. Defaults to 10.

Value

List of ggplot objects containing plots of walks for low-value parameters, high-value parameters, and normalized negative log likelihood.

plot.parameters *Plot of wisp parameters*

Description

Function to make nicely formatted violin (or bar) plots of the fitted wisp parameters, including confidence intervals if stat information is available.

Usage

```
## S3 method for class 'parameters'
plot(
  wisp.results,
  child.lvl = NULL,
  violin = TRUE,
  print.plots = TRUE,
  child.classes = NULL,
  verbose = TRUE
)
```

Arguments

`wisp.results`	List, output of the wisp function.
`child.lvl`	Character string, the child level to be plotted. If NULL, all child levels are plotted.
`violin`	Logical, if TRUE, plots violin plots for each parameter; if FALSE, uses bar plots.
`print.plots`	Logical, if TRUE, prints the plots to the console; if FALSE, only returns a list of plots without printing.
`child.classes`	List, a list of character vectors specifying which child levels to include together in plots. If NULL, all child levels are included in a single plot.
`verbose`	Logical, if TRUE, prints updates about the plotting process.

Value

List of ggplot objects for parameter plots.

plot.ratecount *Plot fitted model and data*

Description

This function takes wisp model results (a fitted line) and observed data (counts) and plots them together for visual comparison. It can also include independent block boundaries for comparison if provided.

Usage

```
## S3 method for class 'ratecount'
plot(
  wisp.results,
  pred.type = ”pred.log”,
  count.type = ”count.log”,
  dim.boundaries = c(),
  print.all = FALSE,
  y.lim = NA,
  count.alpha.none = NA,
  count.alpha.ran = NA,
  pred.alpha.none = NA,
  pred.alpha.ran = NA,
  rans.to.print = c(),
  childs.to.print = c(),
  verbose = TRUE
)
```

Arguments

`wisp.results` List, output of the wisp function.

`pred.type` Character string, the name of the predicted rate column in the count data (e.g., "pred.log" or "pred").

`count.type` Character string, the name of the observed count column in the count data (e.g., "count.log" or "count").

`dim.boundaries` Numeric vector, independent block boundaries to plot for comparison. If empty, the argument is ignored.

`print.all` Logical, if TRUE, prints all plots; if FALSE, only returns plots in list without printing any.

`y.lim` Numeric vector of length 2, limits for the y-axis of the plots. If NA, defaults to automatic limits.

`count.alpha.none` Numeric, transparency for count points when random level is "none". If left NA, defaults to 0.25.

`count.alpha.ran` Numeric, transparency for count points when random level is not "none". If left NA, defaults to 0.25.

`pred.alpha.none` Numeric, transparency for predicted lines when random level is "none". If left NA, defaults to 1.0.

`pred.alpha.ran` Numeric, transparency for predicted lines when random level is not "none". If left NA, defaults to 0.9.

`rans.to.print` Character vector, list of random levels to include on each child plot. If c(), all random levels are included.

`childs.to.print` Character vector, list of child levels to place on their own plot. If c(), all child levels are plotted individually.

`verbose` Logical, if TRUE, prints updates about the plotting process.

Value

List of ggplot objects for rate-count plots.

pvalues.samples *Compute p-values using ecdf from parameter resamples*

Description

This function takes a vector `mu.B` of sampled values of a variable X, and a single observation `mu.obs` of the same variable, and computes the p-value of `mu.obs` using the empirical cumulative distribution function (ecdf) of `mu.B`.

Usage

```
pvalues.samples(mu.B, mu.obs)
```

Arguments

`mu.B`	Numeric vector of sampled values of a variable X, e.g., bootstrapped or MCMC estimates, used to make an empirical cumulative distribution function (ecdf).
`mu.obs`	Numeric value of the observed variable X, e.g., the mean of `mu.B` or an actual observation.

Value

Numeric p-value.

sample.stats *Estimate p-values and confidence intervals from resampled parameters*

Description

Runs statistical analysis on the resampled parameters from the wisp function. It computes p-values and confidence intervals for each parameter, adjusting for multiple comparisons using either the Bonferroni correction or the Holm-Bonferroni method.

Usage

```
sample.stats(
  wisp.results,
  alpha = 0.05,
  Bonferroni = FALSE,
  conv.resamples.only = TRUE,
  verbose = TRUE
)
```

Arguments

`wisp.results` List, output of the wisp function.

`alpha` Numeric value giving significance level for p-values and confidence intervals. Default is 0.05.

`Bonferroni` Logical, if TRUE, uses the Bonferroni correction for multiple comparisons; if FALSE, uses the Holm-Bonferroni method. Default is FALSE.

`conv.resamples.only`
Logical, if TRUE, only resamples with a converged fit are used for statistical analysis; if FALSE, all resamples are used. Default is TRUE.

`verbose` Logical, if TRUE, prints information during the statistical analysis.

Value

Data frame giving, for each parameter, its name, estimate, confidence interval (CI.low, CI.high), p-value, adjusted p-value (p.value.adj), adjusted alpha (alpha.adj), and significance level (significance).

`wisp` *Fit wisp to count data*

Description

This function takes a data frame of wisp variables (as columns) and fits a wisp model to it. Statistical analyses and plots are generated from the fitted model.

Usage

```
wisp(
  count.data,
  variables = list(),
  use.median = FALSE,
  MCMC.settings = list(),
  bootstraps.num = 0,
  converged.resamples.only = TRUE,
  max.fork = 1,
  dim.bounds = c(),
  verbose = TRUE,
  print.child.summaries = TRUE,
  model.settings = list()
)
```

Arguments

`count.data` Data.frame, data to be modeled, with columns for model variables (count, bin, parent, child, ran, fixedeffects), or equivalent variables as specified in the `variables` argument.

`variables` List, names of the columns in `count.data` that correspond to the model variables. The list should contain only (but not necessarily all) named elements: `count`, `bin`, `parent`, `child`, `ran`, and `fixedeffects`.

`use.median` Logical, if TRUE, the median of the resamples is used as the final parameter estimates; if FALSE, the initial fit by L-BFGS is used.

`MCMC.settings` List, settings for the MCMC simulation, including `MCMC.burnin`, `MCMC.steps`, `MCMC.step.size`, `MCMC.prior`, and `MCMC.neighbor.filter`. Default values are provided.

`bootstraps.num` Integer, number of bootstrap resamples to perform. If 0, only MCMC is run.

`converged.resamples.only`
Logical, if TRUE, only resamples with a converged fit are used for statistical analysis; if FALSE, all resamples are used. Applies only to bootstraps.

`max.fork` Integer, maximum number of parallel processes to use for bootstrapping.

`dim.bounds` Numeric vector, block boundaries for plotting in rate-count plots. If empty, the argument is ignored.

`verbose` Logical, if TRUE, prints information during the fitting process.

`print.child.summaries`
Logical, if TRUE, prints summaries of each child level.

`model.settings` List, settings for the C++ model, including `buffer_factor`, `ctol`, `max_penalty_at_distance_fact`
`LROcutoff`, `LROwindow_factor`, `rise_threshold_factor`, `max_evals`, `rng_seed`, and `warp_precision`. Default values are provided.

Value

List giving the results of the fitted model, including: `model.component.list`, `count.data.summed`, `fitted.parameters`, `gamma.disperson`, `param.names`, `fix`, `treatment`, `grouping.variables`, `param.idx0`, `settings`, `sample.params`, `sample.params.bs`, `sample.params.MCMC`, `diagnostics.bs`, `diagnostics.MCMC`, `stats`, and `plots`

`wisp.sigmoid` *Wisp sigmoid function, implemented in R*

Description

This function provides an R implementation of the wisp sigmoid function. It assumes the parameters Rt, tslope, and tpoint have already been warped by `wisp.warp`.

Usage

```
wisp.sigmoid(x, Rt, tslope, tpoint)
```

Arguments

`x` Numeric, input spatial position, either a scalar, vector, or 2D array/matrix

`Rt` Numeric vector, rate parameters for the wisp function. Degree of the wisp model will be length of this vector minus 1.

`tslope` Numeric vector, slope scalars for the wisp function. Must be one less than the length of Rt.

`tpoint` Numeric vector, transition points for the wisp function. Must be one less than the length of Rt.

Value

Numeric vector or matrix, wisp sigmoid values for given input. Returned object will have the same dimensions as input `x`.

`wisp.warp` *Wisp warping function, implemented in R*

Description

This function provides an R implementation of the warping function used to warp model components before input into `wisp.sigmoid`.

Usage

```
wisp.warp(z, b, w)
```

Arguments

`z`	Numeric vector or matrix, values to warp. Scalar values fine as well. Matrix must be 2D.
`b`	Numeric, warping bound (must be a single value).
`w`	Numeric, warping factor (must be a single value).

Value

Numeric vector or matrix, warped values. Returned object will have the same dimensions as input `z`.

Index