Package: WSPsignal (via r-universe)

July 9, 2026

Title Weibull Shape Parameter Tests for Signal Detection

Version 1.0.0

Description Implementation of Bayesian and frequentist Weibull Shape Parameter (WSP) tests for signal detection in pharmacovigilance based on right-censored time-to-event data to flag associations between drugs and adverse events. The WSP test is based on the assumption of constant hazard reflected by a Weibull type distribution with shape parameters equal to one. Based on the shape parameter estimates (posterior distribution or point estimate), the WSP test method performs a hypothesis test on each shape parameter and combines them to a decision on the presence of a signal. Methods described in Sauzet and Cornelius (2022) <doi:10.3389/fphar.2022.889088>, Sauzet et al. (2024) <doi:10.1007/s40264-024-01460-2>, and Dyck and Sauzet (2025) <doi:10.48550/arXiv.2412.05463>.

License MIT + file LICENSE

Encoding UTF-8

Biarch true

Depends R (>= 3.5.0)

Imports methods, Rcpp (>= 0.12.0), RcppParallel (>= 5.0.1), rstan (>= 2.18.1), rstantools (>= 2.4.0), Rdpack, HDInterval, ROCR, stats, graphics, magrittr, dplyr, tidyr, ggplot2, survival, furrr

LinkingTo BH (>= 1.66.0), Rcpp (>= 0.12.0), RcppEigen (>= 0.3.3.3.0), RcppParallel (>= 5.0.1), rstan (>= 2.18.1), StanHeaders (>= 2.18.0)

SystemRequirements GNU make

RdMacros Rdpack

LazyData true

Config/roxygen2/version 8.0.0

NeedsCompilation yes

Author Julia Dyck [cre], Julia Dyck [aut], Odile Sauzet [aut]

Maintainer Julia Dyck <j.dyck@uni-bielefeld.de>

Config/pak/sysreqs make

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

Date/Publication 2026-07-09 08:54:33 UTC

RemoteUrl https://github.com/cran/WSPsignal

RemoteRef HEAD

RemoteSha 58165735aa7acf22e485b5b01996f410852c1720

Contents

`WSPsignal-package` *The 'WSPsignal' package.*

Description

An R package to perform Bayesian and frequentist Weibull Shape Parameter (WSP) tests for signal detection.

Author(s)

Maintainer: Julia Dyck <j.dyck@uni-bielefeld.de>

Authors:

- Julia Dyck
- Odile Sauzet

References

Sauzet O, Cornelius V (2022). "Generalised weibull model-based approaches to detect non-constant hazard to signal adverse drug reactions in longitudinal data." *Frontiers in Pharmacology*. doi:10.3389/fphar.2022.889088. https://pubmed.ncbi.nlm.nih.gov/36081935/.

Sauzet O, Dyck JA, Cornelius V (2024). "Optimal Significance Levels and Sample Sizes for Signal Detection Methods Based on Non-constant Hazards." *Drug Safety*. ISSN 1179-1942. doi:10.1007/s40264024014602. https://pub.uni-bielefeld.de/record/2991311.

Dyck J, Sauzet O (2025). "The BPgWSP test: a Bayesian Weibull Shape Parameter signal detection test for adverse drug reactions." preprint, 2412.05463, https://arxiv.org/abs/2412.05463.

Stan Development Team (NA). RStan: the R interface to Stan. R package version 2.32.6. https://mc-stan.org

---

`bwsp_model` *Fit Bayesian model to time-to-event data*

Description

Fits a Bayesian model to time-to-event (tte) data for the purpose of performing Weibull shape parameter signal detection tests with `bwsp_test`.

Usage

```
bwsp_model(datstan, chains = 4, iter = 11000, warmup = 1000)
```

Arguments

`datstan`	named list of data for the stanmodel; output of `tte2priordat`
`chains`	number of Markov chains to run
`iter`	total number of iterations per chain (including warmup)
`warmup`	number of warmup iterations per chain

Details

The function applies the sampling command with the No U-Turn sampler to fit a Bayesian model to tte data.

The posterior is proportional to the likelihood times the prior. The likelihood used in ML estimation is

$$\mathcal{L}(t) = \prod_{i=1}^{N} S(t_i)^{1-d_i} \cdot f(t_i)^{d_i}$$

with $S(t)$ being the survival function of the chosen distribution, $f(t)$ the density (Nikulin et al. 2016), and (t_i, d_i) the (right-censored) tte observations.

Prior and tte distribution are specified in the previous data preparation step with function tte2priordat.

Value

A list with components:

- fit: fitted model object; a stanfit object for "w" and "pgw", or a list of two stanfit objects ($uncens, $cens) for "dw"
- args_list: list of model specifications, including the chosen time-to-event distribution and prior settings

References

Nikulin M, Wu HI, others (2016). *The Cox model and its applications*. Springer.

Examples

```
head(tte)

# prep the data
# we formalize a prior belief (here "no association
# between drug and event", therefore prior mean = 1 for shape parameter)
# and reformat our tte data to fit the model in the following
dat_list = tte2priordat(dat = tte,   # reformat the data
                        tte.dist = "w",
                        prior.dist = "ll",
                        scale.mean = 1,
                        scale.sd = 10,
                        shape.mean = 1,
                        shape.sd = 10)

# model fitting
mod = bwsp_model(datstan = dat_list,      # fit the model
                 chains = 4,
                 iter = 110,             # (posterior sample is
                 warmup = 10)            # small for demo purpose)
mod$fit
```

`bwsp_test` *Bayesian Weibull Shape Parameter Test*

Description

Bayesian Weibull Shape Parameter (BWSP) test of the constant hazard (null-)hypothesis, based on the shape parameter(s) of Weibull family of distributions.

Usage

```
bwsp_test(
  mod.output,
  cred.level = 0.8,
  ci.type = "HDI",
  sensitivity.option = 2
)
```

Arguments

`mod.output`	model output resulting from `bwsp_model`
`cred.level`	numeric or vector of credibility levels; default is 0.8
`ci.type`	character indicating whether to extract an equal tailed interval (`"ETI"`) or highest posterior density interval (`"HDI"`) as posterior credibility interval (CI) for the BWSP test; default is `"HDI"`
`sensitivity.option`	numeric value out of `1,2,3`; combination rule to deduct a binary outcome (signal/no signal) from one or two shape parameter tests; default is `sensitivity.option = 2` (see details)

Value

binary vector, 0 if H_0 is accepted (no signal), 1 if H_1 is accepted (signal)

Test concept

The BWSP test is based on the principle of (non-)constant hazard (Cornelius et al. 2012) which associates a constant hazard function with the absence of a drug-adverse event association and a non-constant hazard with the presence of a drug-adverse event association.

This can be formalized as the following hypotheses depending on the underlying model:

	H_0	H_1
hypothesis	constant hazard function	non-constant hazard function
under Weibull model	$\nu = 1$	$\nu \neq 1$
under double Weibull model	$\nu_1 = 1$ and $\nu_2 = 1$	$\nu_1 \neq 1$ or $\nu_2 \neq 1$
under Power generalized Weibull model	$\nu = 1$ and $\gamma = 1$	$\nu \neq 1$ or $\gamma \neq 1$

Bayesian test components

Information on the Bayesian variant of the Power Generalized Weibull (PGW) shape parameter test can be found in Dyck and Sauzet (2025). The same concept applies to the construction of the Bayesian Weibull and double Weibull shape parameter test.

The region of practical equivalence (ROPE) represents the expected parameter value under H_0. The posterior credibility interval(s) (CI) represent the posterior distribution of each shape parameter. For the ROPE, the function sets up an equal-tailed interval (ETI)

$$[q_{(1-\alpha)/2}, q_{(1+\alpha)/2}]$$

based on the quantiles q of the shape parameters' prior distributions under H_0 at a chosen credibility level $1 - \alpha$.

For the posterior CI, the function calculates either an ETI at the same credibility level obtained from the empirical quantiles of the posterior distribution per shape parameter or a highest density interval (HDI, Kruschke (2015))

$$HDI(\nu) = \{\nu \mid p_1(\nu) \geq w\} \text{ with } w \in [0,1] \text{ such that } \int_{\nu \mid p_1(\nu) \geq w} p_1(\nu|t)\, d\nu = 1 - \alpha$$

where ν is one of the shape parameters, p_1 it's posterior density and t the time variable.

The CI+ROPE test (Kruschke 2018) checks the relationship between ROPE and posterior CI leading to either acceptance, rejection or no decision regarding the null hypothesis for a single shape parameter. Sensitivity options to generate a binary outcome, i.e. a signal or not, from CI+ROPE test results based on one (in case of `"w"`) or two (in case of `"dw"`, `"pgw"`) shape parameters are:

HDI+ROPE outcome for shape_1	HDI+ROPE outcome for shape_2	combination rule (`sensitivity.option = 1`)	combination rule (`sensitivity.option = 2`)	combination rule (`sensitivity.option = 3`)
rejection	(none)	signal	signal	signal
acceptance	(none)	-	-	-
no decision	(none)	signal	-	-
rejection	rejection	signal	signal	signal
acceptance	rejection	signal	-	-
rejection	acceptance	signal	-	-
acceptance	acceptance	-	-	-
no decision	rejection	signal	signal	-
no decision	acceptance	-	-	-
rejection	no decision	signal	signal	-
acceptance	no decision	-	-	-
no decision	no decision	signal	-	-

The hypotheses as stated above (see test concept) are implemented in `sensitivity.option = 1` whereas `sensitivity.option = 2` and `sensitivity.option = 3` lead to a signal in fewer cases.

More details on the CI+ROPE test, recommendations for interval specifications and the combination rules can be found in Kruschke (2018) and Dyck and Sauzet (2025).

References

Cornelius VR, Sauzet O, Evans SJ (2012). "A signal detection method to detect adverse drug reactions using a parametric time-to-event model in simulated cohort data." *Drug safety*, **35**, 599–610.

Dyck J, Sauzet O (2025). "The BPgWSP test: a Bayesian Weibull Shape Parameter signal detection test for adverse drug reactions." preprint, 2412.05463, https://arxiv.org/abs/2412.05463.

Kruschke J (2015). *Doing Bayesian Data Analysis (Second Edition)*. Academic Press, Boston.

Kruschke JK (2018). "Rejecting or Accepting Parameter Values in Bayesian Estimation." *Advances in Methods and Practices in Psychological Science*, **1**(2), 270-280. doi:10.1177/2515245918771304. https://doi.org/10.1177/2515245918771304.

Examples

```
#### Exemplary conduction of a test from data and prior to test result:

# under weibull model:

# 1. prior specification
# we formalize a prior belief (here "no association
# between drug and event", therefore prior mean = 1 for shape parameter)
# and reformat our tte data to fit the model in the following
dat_list = tte2priordat(dat = tte,   # reformat the data
                        tte.dist = "w",
                        prior.dist = "ll",
                        scale.mean = 1,
                        scale.sd = 10,
                        shape.mean = 1,
                        shape.sd = 10)

# 2. model fitting
fit = bwsp_model(datstan = dat_list,      # fit the model
                 chains = 4,
                 iter = 110,             # (posterior sample is
                 warmup = 10)            # small for demo purpose)
fit$fit

# 3. BWSP test
bwsp_test(mod.output = fit,
          cred.level = 0.8,
          ci.type = "HDI",
          sensitivity.option = 2)

# under pgw model:

# 1. prior specification
# we formalize a prior belief (here "no association
# between drug and event", therefore prior mean = 1 for both shape parameters)
# and reformat our tte data to fit the model in the following
dat_list = tte2priordat(dat = tte,          # reformat the data
```

```
                        tte.dist = "pgw",
                        prior.dist = "ll",
                        scale.mean = 1,
                        scale.sd = 10,
                        shape.mean = 1,
                        shape.sd = 10,
                        powershape.mean = 1,
                        powershape.sd = 10)

# 2. model fitting
fit = bwsp_model(datstan = dat_list,     # fit the model
                 chains = 4,
                 iter = 110,             # (posterior sample
                 warmup = 10)            # is small for demo purpose)

# 3. BWSP test
bwsp_test(mod.output = fit,
          cred.level = 0.8,
          ci.type = "HDI",
          sensitivity.option = 2)
```

`eval.calc_perf` *Compute performance metrics for WSP test configurations*

Description

Computes performance metrics for all the specified Weibull Shape Parameter (WSP) test configurations across simulated scenarios. The output provides the base for a ranking of tests (see `eval.rank_auc`).

Usage

```
eval.calc_perf(pc_list)
```

Arguments

`pc_list`	list of simulation parameters generated with `sim.setup_sim_pars`

Details

Based on the merged simulation results obtained with `sim.merge_results`, the function performs WSP tests for all specified model and test configurations. Bayesian WSP tests depend on the combination of time-to-event (tte) distribution, prior model specification, posterior credibility interval (CI) type, credibility level and sensitivity option (see `bwsp_test`). Frequentist WSP tests depend on the tte distribution and confidence level (see `fwsp_test`).

Given binary test results the function computes the following performance measures:

- False positive rate:
$$fpr = \frac{FP}{FP + TN}$$
- True positive rate (sensitivity, recall):
$$tpr = \frac{TP}{TP + FN}$$
- False negative rate:
$$fnr = \frac{FN}{TP + FN}$$
- True negative rate (specificity):
$$tnr = \frac{TN}{FP + TN}$$

with FP being the number of false positive cases, TN the number of true negative cases, TP the number of true positive cases and FN the number of false negative cases among simulation repetitions, as well as

- Area under the ROC curve (AUC):
  The AUC is the area under the receiver operating characteristic (ROC) graph (Fawcett 2004). Here, the ROC curve with one threshold based on equal numbers of ADR-positive and control scenarios is computed using the `performance` function.

Value

A data frame containing one row per ADR–positive scenario, WSP model and test configuration, and corresponding performance measurements in additional columns, namely the auc, fpr, tpr, fnr and tnr.

Scenarios with incomplete number of repetitions return `NA` for performance metrics. Frequentist WSP tests return `NA` for scenario/model characteristics that are only relevant for Bayesian WSP test specification.

References

Fawcett T (2004). "ROC graphs: Notes and practical considerations for researchers." *Machine learning*, **31**(1), 1–38.

Examples

```
# The package ships with a small precomputed toy simulation study with small
# numbers of repetitions (rep) and posterior sample sizes (stanmod.iter, stanmod.warmup).
# The simulation parameters below match the shipped example results.

#### prep:
toy_path <- system.file("extdata", "toysim", package = "WSPsignal")

# setup prior template
fp_list <- sim.priors_template(tte.dist = c("w", "pgw"), prior.sds = 10)
```

```
# fill in prior template with prior means
fp_list$w[,2] <- c(1, 1, 180, 300)
fp_list$w[,3] <- c(1, 0.207, 1, 4)

fp_list$pgw[,2] <- c(1, 1, 20, 300)
fp_list$pgw[,3] <- c(1, 0.207, 5.5, 4)
fp_list$pgw[,4] <- c(1, 1, 14, 1)

# recreate simulation settings used for the toy example
pc_list <- sim.setup_sim_pars(
  N = 500,
  br = 0.1,
  adr.rate = c(0, 1),
  adr.relsd = 0.05,
  study.period = 365,
  est.approach = c("f", "b"),
  tte.dist = c("w", "pgw"),
  prior.dist = "ll",
  fitpars.list = fp_list,
  post.ci.type = c("ETI", "HDI"),
  cred.level = seq(0.5, 0.9, by = 0.05),
  sensitivity.option = 1:3,
  reps = 6,
  batch.size = 3,
  resultpath = toy_path,
  stanmod.iter = 1100,
  stanmod.warmup = 100
)


#### compute performance metrics based on merged simulation results

perf = eval.calc_perf(pc_list)
head(perf, 10)
```

`eval.eff_sample_sizes` *Evaluate effective sample sizes*

Description

Only for Bayesian estimation approach. Summarizes effective sample sizes of the stan models fitted optionally grouped by one or multiple model specifications. This helps assess which of the modelling choices is suitable for CI+ROPE testing (along with other diagnostics such as `eval.non_conv_cases` and `eval.execution_times`).

Usage

```
eval.eff_sample_sizes(
  pc_list,
```

```
  group.by = c("tte.dist", "prior.dist", "prior.sd"),
  threshold = 10000,
  verbose = TRUE
)
```

Arguments

pc_list	list of simulation parameters generated with sim.setup_sim_pars
group.by	character vector specifying grouping variables; must be a subset of c("tte.dist", "prior.dist", "prior.sd")
threshold	numeric threshold for effective sample size acceptable for HDI+ROPE testing (10000 by default as recommended by Kruschke (2015))
verbose	logical; if TRUE (default), summary statistics and plot are printed to the console

Value

A list with summary statistics ($summary), a ggplot2 object ($plot), and the data ($df) on which summary and plot are based.

References

Kruschke J (2015). *Doing Bayesian Data Analysis (Second Edition).* Academic Press, Boston.

See Also

eval.non_conv_cases, eval.execution_times

Examples

```
# The package ships with a small precomputed toy simulation study with small
# numbers of repetitions (rep) and posterior sample sizes (stanmod.iter, stanmod.warmup).
# The simulation parameters below match the shipped example results.

#### prep:
toy_path <- system.file("extdata", "toysim", package = "WSPsignal")

# setup prior template
fp_list <- sim.priors_template(tte.dist = c("w", "pgw"), prior.sds = 10)

# fill in prior template with prior means
fp_list$w[,2] <- c(1, 1, 180, 300)
fp_list$w[,3] <- c(1, 0.207, 1, 4)

fp_list$pgw[,2] <- c(1, 1, 20, 300)
fp_list$pgw[,3] <- c(1, 0.207, 5.5, 4)
fp_list$pgw[,4] <- c(1, 1, 14, 1)

# recreate simulation settings used for the toy example
pc_list <- sim.setup_sim_pars(
  N = 500,
```

```
  br = 0.1,
  adr.rate = c(0, 1),
  adr.relsd = 0.05,
  study.period = 365,
  est.approach = c("f", "b"),
  tte.dist = c("w", "pgw"),
  prior.dist = "ll",
  fitpars.list = fp_list,
  post.ci.type = c("ETI", "HDI"),
  cred.level = seq(0.5, 0.9, by = 0.05),
  sensitivity.option = 1:3,
  reps = 6,
  batch.size = 3,
  resultpath = toy_path,
  stanmod.iter = 1100,
  stanmod.warmup = 100
)

##### summarize effective sample sizes only by tte distribution (only varying factor)
n_eff = eval.eff_sample_sizes(pc_list, group.by = "tte.dist")
```

`eval.execution_times` *Evaluate execution times*

Description

Only for Bayesian estimation approach. Summarizes execution times of the models fitted optionally grouped by one or multiple model specifications. This helps assess which of the modelling choices is suitable for CI+ROPE testing (along with other diagnostics such as `eval.non_conv_cases` and `eval.eff_sample_sizes`).

Usage

```
eval.execution_times(
  pc_list,
  group.by = c("tte.dist", "prior.dist", "prior.sd"),
  verbose = TRUE
)
```

Arguments

`pc_list`	list of simulation parameters generated with `sim.setup_sim_pars`
`group.by`	character vector specifying grouping variables; must be a subset of `c("tte.dist", "prior.dist", "prior.sd")`
`verbose`	logical; if `TRUE` (default), summary statistics and plot are printed to the console

Details

Calculations are based on the stored result file obtained with sim.merge_results.

Value

A list with summary statistics ($summary), a ggplot2 object ($plot), and the data ($df) on which summary and plot are based.

See Also

eval.non_conv_cases, eval.eff_sample_sizes

Examples

```
# The package ships with a small precomputed toy simulation study with small
# numbers of repetitions (rep) and posterior sample sizes (stanmod.iter, stanmod.warmup).
# The simulation parameters below match the shipped example results.

#### prep:
toy_path <- system.file("extdata", "toysim", package = "WSPsignal")

# setup prior template
fp_list <- sim.priors_template(tte.dist = c("w", "pgw"), prior.sds = 10)

# fill in prior template with prior means
fp_list$w[,2] <- c(1, 1, 180, 300)
fp_list$w[,3] <- c(1, 0.207, 1, 4)

fp_list$pgw[,2] <- c(1, 1, 20, 300)
fp_list$pgw[,3] <- c(1, 0.207, 5.5, 4)
fp_list$pgw[,4] <- c(1, 1, 14, 1)

# recreate simulation settings used for the toy example
pc_list <- sim.setup_sim_pars(
  N = 500,
  br = 0.1,
  adr.rate = c(0, 1),
  adr.relsd = 0.05,
  study.period = 365,
  est.approach = c("f", "b"),
  tte.dist = c("w", "pgw"),
  prior.dist = "ll",
  fitpars.list = fp_list,
  post.ci.type = c("ETI", "HDI"),
  cred.level = seq(0.5, 0.9, by = 0.05),
  sensitivity.option = 1:3,
  reps = 6,
  batch.size = 3,
  resultpath = toy_path,
  stanmod.iter = 1100,
  stanmod.warmup = 100
)
```

```
#### summarize execution times only by tte distribution (only varying factor)
ex_times = eval.execution_times(pc_list, group.by = "tte.dist")
```

`eval.non_conv_cases` *Evaluate number of non-convergence cases*

Description

Only for Bayesian estimation approach. Summarizes the number of planned vs. not successfully run simulations optionally grouped by one or multiple model specifications. This helps assess which of the modelling choices is suitable for CI+ROPE testing (along with other diagnostics such as `eval.execution_times` and `eval.eff_sample_sizes`).

Usage

```
eval.non_conv_cases(
  pc_list,
  group.by = c("tte.dist", "prior.dist", "prior.sd"),
  verbose = TRUE
)
```

Arguments

`pc_list`	list of simulation parameters generated with `sim.setup_sim_pars`
`group.by`	character vector specifying grouping variables; must be a subset of `c("tte.dist", "prior.dist", "prior.sd")`
`verbose`	logical; if `TRUE` (default), output is printed to the console

Details

Calculations are based on the stored result file obtained with `sim.merge_results`.

Value

A data frame with the following columns:

- `tte.dist`: The tte distribution as grouping factor (if selected).
- `prior.dist`: The prior distribution as grouping factor (if selected).
- `prior.sd`: The prior standard deviation as grouping factor (if selected).
- `total.planned`: The total number of planned repetitions in this group.
- `total.notrun`: The total number of repetitions that were not run in this group.
- `prop.notrun`: The proportion of repetitions that were not run in this group.

See Also

eval.execution_times, eval.eff_sample_sizes

Examples

```
# The package ships with a small precomputed toy simulation study with small
# numbers of repetitions (rep) and posterior sample sizes (stanmod.iter, stanmod.warmup).
# The simulation parameters below match the shipped example results.

#### prep:
toy_path <- system.file("extdata", "toysim", package = "WSPsignal")

# setup prior template
fp_list <- sim.priors_template(tte.dist = c("w", "pgw"), prior.sds = 10)

# fill in prior template with prior means
fp_list$w[,2] <- c(1, 1, 180, 300)
fp_list$w[,3] <- c(1, 0.207, 1, 4)

fp_list$pgw[,2] <- c(1, 1, 20, 300)
fp_list$pgw[,3] <- c(1, 0.207, 5.5, 4)
fp_list$pgw[,4] <- c(1, 1, 14, 1)

# recreate simulation settings used for the toy example
pc_list <- sim.setup_sim_pars(
  N = 500,
  br = 0.1,
  adr.rate = c(0, 1),
  adr.relsd = 0.05,
  study.period = 365,
  est.approach = c("f", "b"),
  tte.dist = c("w", "pgw"),
  prior.dist = "ll",
  fitpars.list = fp_list,
  post.ci.type = c("ETI", "HDI"),
  cred.level = seq(0.5, 0.9, by = 0.05),
  sensitivity.option = 1:3,
  reps = 6,
  batch.size = 3,
  resultpath = toy_path,
  stanmod.iter = 1100,
  stanmod.warmup = 100
)


#### summarize non-convergence cases only by tte distribution (only varying factor)

non_conv = eval.non_conv_cases(pc_list, group.by = "tte.dist")
```

`eval.rank_auc` *Ranking of WSP test configurations*

Description

Ranks all model and test specifications grouped by simulation scenarios in terms of the corresponding area under the curve (AUC) value for Weibull Shape Parameter (WSP) tests.

Usage

```
eval.rank_auc(
  perf,
  test.type.subset = c("bwsp", "fwsp"),
  tte.dist.subset = c("w", "dw", "pgw"),
  prior.dist.subset = c("fg", "fl", "gg", "ll"),
  prior.sd.subset = NULL,
  verbose = TRUE
)
```

Arguments

`perf`	data frame containing performance results for WSP tests returned by `eval.calc_perf`
`test.type.subset`	character to filter for Bayesian and frequentist WSP test types to be considered in the ranking; must be a subset of `c("bwsp", "fwsp")`
`tte.dist.subset`	character to filter for the time-to-event (tte) distributions considered in the ranking, must be a subset of `c("w", "dw", "pgw")`
`prior.dist.subset`	character to filter for the prior distribution (relevant only for BWSP tests), must be a subset of `c("fg", "fl", "gg", "ll")`
`prior.sd.subset`	numeric to filter for the prior standard deviation (relevant only for BWSP tests), must be a subset of prior.sds considered in the simulation
`verbose`	logical; if `TRUE` (default), `$rank.tab` of output list is printed to the console

Details

For definitions of the performance metrics `AUC`, `FPR`, `TPR`, `FNR` and `TNR` returned in output, see the details section of `eval.calc_perf`.

The filter mechanism enables filtering for a subset of test specifications. This is helpful for example when tte distributions, prior distributions or an estimation approach are no longer under consideration for instance after inspecting the model diagnostics with `eval.execution_times`, `eval.non_conv_cases` and `eval.eff_sample_sizes`.

Value

A list containing

- `$rank.tab`: Ranking of fit and WSP test specifications according to AUC averaged over all sample scenarios (for BWSP tests given a correct specification of prior belief)
- `$effect.of.N`: Effect of sample size on AUC for the optimal fit and WSP test (for BWSP tests given a correct specification of prior belief)
- `$effect.of.br`: Effect of background rate on AUC for the optimal fit and WSP test (for BWSP tests given a correct specification of prior belief)
- `$effect.of.adr.rate`: Effect of ADR rate on AUC for the optimal fit and WSP test (for BWSP tests given a correct specification of prior belief)
- `$effect.of.adr.when`: Effect of true expected event times on AUC for the optimal fit and WSP test (for BWSP tests given a correct specification of prior belief)
- `$effect.of.adr.relsd`: Effect of relative standard deviation of event time on AUC for the optimal fit and WSP test (for BWSP tests given a correct specification of prior belief)
- `$effect.of.dist.prior.to.truth`: Effect of distance of prior belief to true `adr.when` on AUC for the optimal fit and WSP test (only BWSP)

Examples

```
# The package ships with a small precomputed toy simulation study with small
# numbers of repetitions (rep) and posterior sample sizes (stanmod.iter, stanmod.warmup).
# The simulation parameters below match the shipped example results.

#### prep:
toy_path <- system.file("extdata", "toysim", package = "WSPsignal")

# setup prior template
fp_list <- sim.priors_template(tte.dist = c("w", "pgw"), prior.sds = 10)

# fill in prior template with prior means
fp_list$w[,2] <- c(1, 1, 180, 300)
fp_list$w[,3] <- c(1, 0.207, 1, 4)

fp_list$pgw[,2] <- c(1, 1, 20, 300)
fp_list$pgw[,3] <- c(1, 0.207, 5.5, 4)
fp_list$pgw[,4] <- c(1, 1, 14, 1)

# recreate simulation settings used for the toy example
pc_list <- sim.setup_sim_pars(
  N = 500,
  br = 0.1,
  adr.rate = c(0, 1),
  adr.relsd = 0.05,
  study.period = 365,
  est.approach = c("f", "b"),
  tte.dist = c("w", "pgw"),
  prior.dist = "ll",
  fitpars.list = fp_list,
```

```
  post.ci.type = c("ETI", "HDI"),
  cred.level = seq(0.5, 0.9, by = 0.05),
  sensitivity.option = 1:3,
  reps = 6,
  batch.size = 3,
  resultpath = toy_path,
  stanmod.iter = 1100,
  stanmod.warmup = 100
)

# compute performance metrics based on merged simulation results
perf = eval.calc_perf(pc_list)

#### rank WSP test conficurations

rank = eval.rank_auc(perf)
```

`eval.roc_curve` *Plot ROC curves for top WSP test specifications*

Description

Plots receiver operating characteristic (ROC) curves for the top-ranked test specifications.

Usage

```
eval.roc_curve(rank.tab, n = 10, verbose = TRUE)
```

Arguments

`rank.tab`	data frame of ranked test specifications obtained from `eval.rank_auc` (`output$rank.tab`)
`n`	number of top-ranked test specifications to plot (10 by default)
`verbose`	logical; if `TRUE` (default), plot is printed to the console

Details

The function returns the receiver ROC curves for the top n WSP test configurations based on the ranking returned by `eval.rank_auc` by plotting the true positive rate (TPR) on the y-axis against the false positive rate (FPR) on the x-axis (Fawcett 2004). Here, we use the ROC curve with one threshold based on equal numbers of ADR-positive and control scenarios.

For definitions of the performance metrics `AUC`, `FPR`, `TPR`, `FNR` and `TNR` returned in printed output, see the details section of `eval.calc_perf`.

The ggplot output can be adjusted to individual needs by adding `ggplot2` layers to the output.

Value

A ggplot object displaying ROC curves with shaded AUC regions.

References

Fawcett T (2004). "ROC graphs: Notes and practical considerations for researchers." *Machine learning*, **31**(1), 1–38.

Examples

```
# The package ships with a small precomputed toy simulation study with small
# numbers of repetitions (rep) and posterior sample sizes (stanmod.iter, stanmod.warmup).
# The simulation parameters below match the shipped example results.

#### prep:
toy_path <- system.file("extdata", "toysim", package = "WSPsignal")

# setup prior template
fp_list <- sim.priors_template(tte.dist = c("w", "pgw"), prior.sds = 10)

# fill in prior template with prior means
fp_list$w[,2] <- c(1, 1, 180, 300)
fp_list$w[,3] <- c(1, 0.207, 1, 4)

fp_list$pgw[,2] <- c(1, 1, 20, 300)
fp_list$pgw[,3] <- c(1, 0.207, 5.5, 4)
fp_list$pgw[,4] <- c(1, 1, 14, 1)

# recreate simulation settings used for the toy example
pc_list <- sim.setup_sim_pars(
  N = 500,
  br = 0.1,
  adr.rate = c(0, 1),
  adr.relsd = 0.05,
  study.period = 365,
  est.approach = c("f", "b"),
  tte.dist = c("w", "pgw"),
  prior.dist = "ll",
  fitpars.list = fp_list,
  post.ci.type = c("ETI", "HDI"),
  cred.level = seq(0.5, 0.9, by = 0.05),
  sensitivity.option = 1:3,
  reps = 6,
  batch.size = 3,
  resultpath = toy_path,
  stanmod.iter = 1100,
  stanmod.warmup = 100
)

# compute performance metrics based on merged simulation results
perf = eval.calc_perf(pc_list)

# rank WSP test conficurations
rank = eval.rank_auc(perf, verbose = FALSE)

#### plot ROC curves for top WSP test specifications
```

```
roc = eval.roc_curve(rank$rank.tab)
```

`fwsp_model` *Fit frequentist model to time-to-event data*

Description

Fits a frequentist model to time-to-event (tte) data via maximum likelihood (ML) estimation.

Usage

```
fwsp_model(dat, tte.dist = c("dw"))
```

Arguments

`dat`	data frame or matrix with time information in first column and event information (binary status) in second column
`tte.dist`	character specifying the distribution for the model out of `"w"`, `"dw"`, `"pgw"` (see details)

Details

The model can be a Weibull (`"w"`), a double Weibull (`"dw"`, estimating two Weibull models - one to the data as is and one to the data censored at mid of observation period), or a power generalized Weibull (`"pgw"`) model.

The likelihood used in ML estimation is

$$\mathcal{L}(t) = \prod_{i=1}^{N} S(t_i)^{1-d_i} \cdot f(t_i)^{d_i}$$

with $S(t)$ being the survival function of the chosen distribution, $f(t)$ the density (Nikulin et al. 2016), and (t_i, d_i) the (right-censored) tte observations.

For the estimation of the Weibull models(`"w"`, `"dw"`), the `survreg` function (with no covariates) is used.

The `"pgw"` model is estimated by numerically minimizing the corresponding negative log-likelihood function with `nlm`.

Value

A list with components depending on `tte.dist`:

- `estimates`: data frame of parameter estimates in standard parametrization (scale, shape, powershape)
- `fit`: fitted model object (`summary.survreg` for `"w"`, a list of two `summary.survreg` objects for `"dw"`, and an `nlm` output list for `"pgw"`)
- `tte.dist`: character indicating the fitted tte distribution

References

Nikulin M, Wu HI, others (2016). *The Cox model and its applications*. Springer.

Examples

```
head(tte)
fwsp_model(tte, tte.dist = "w") # Weibull model
fwsp_model(tte, tte.dist = "dw") # double Weibull model
fwsp_model(tte, tte.dist = "pgw") # power generalized Weibull model
```

---

fwsp_test *Frequentist Weibull Shape Parameter Test*

Description

Frequentist Weibull Shape Parameter (FWSP) test of the constant hazard (null-)hypothesis, based on the shape parameter(s) of Weibull family of distributions.

Usage

```
fwsp_test(mod.output, cred.level = 0.96)
```

Arguments

mod.output	model output resulting from fwsp_model
cred.level	numeric or vector of confidence levels (i.e. 1 - significance level) for the test(s) to be performed

Details

This function tests the null hypothesis that the shape parameter(s) of the Weibull family distribution are equal to one. The distribution specific definition of the null and alternative hypotheses can be seen in Sauzet and Cornelius (2022).

For the "w" and "dw" case, the model output is a summary of a survival::Survreg outcome which provides $ln(1/\nu)$ as transform of the shape parameter estimate ν. The transform $ln(1/\nu) = 0$ under the null hypothesis $\nu = 1$. The shape parameter test is performed on the transform equivalent to performing the test based on the shape parameter itself.

For the "pgw" case, the shape parameter test is performed on the logarithmized parameter estimates, i.e. fwsp_test tests the null hypothesis that the logarithm of the shape parameters of the power generalized Weibull distribution are equal to zero based on the shape estimates and their estimated standard errors extracted from the estimated Hessian matrix. Issues with standard error calculation from the estimated Hessian matrix may lead to a NA test results which are then transformed to no signal (0) following Sauzet and Cornelius (2022).

Value

binary vector, 0 if H_0 is accepted (no signal), 1 if H_0 is rejected (signal)

References

Sauzet O, Cornelius V (2022). "Generalised weibull model-based approaches to detect non-constant hazard to signal adverse drug reactions in longitudinal data." *Frontiers in Pharmacology*. doi:10.3389/fphar.2022.889088. https://pubmed.ncbi.nlm.nih.gov/36081935/.

Examples

```
# fit a model
mod = fwsp_model(dat = tte)
mod
# perform the shape parameter test at credibility level 0.95
# or significance level 0.05
fwsp_test(mod.output = mod, cred.level = 0.95)
```

---

muscu *Simulated musculoskeletal pain time-to-event dataset*

Description

A simulated time-to-event (tte) dataset on time to musculoskeletal pain from first bisphosphonate intake.

Usage

```
muscu
```

Format

A data frame with 19 777 rows and 2 variables:

time event time (in days) if an event was observed or censoring time (365 days) if no event was observed,

status event status; 1 if an event was observed, 0 if no event was observed.

Details

The data was generated using `sim.datagen_tte(genpar = c(19777, 0.01, 0.89, 160/365, 0.1, 365))` with parameters derived from the case study presented in Dyck and Sauzet (2025).

References

Dyck J, Sauzet O (2025). "The BPgWSP test: a Bayesian Weibull Shape Parameter signal detection test for adverse drug reactions." preprint, 2412.05463, https://arxiv.org/abs/2412.05463.

See Also

sim.datagen_tte

---

muscu2 *Simulated musculoskeletal pain time-to-event dataset*

Description

A simulated time-to-event (tte) dataset on time to musculoskeletal pain from first bisphosphonate intake.

Usage

```
muscu2
```

Format

A data frame with 1 208 rows and 2 variables:

time event time (in days) if an event was observed or censoring time (365 days) if no event was observed,

status event status; 1 if an event was observed, 0 if no event was observed.

Details

The data was generated using `sim.datagen_tte(genpar = c(1208, 0.01, 0.5, 100/365, 0.1, 365))`. Data generation was guided by the data presented in the case study in Dyck and Sauzet (2025).

References

Dyck J, Sauzet O (2025). "The BPgWSP test: a Bayesian Weibull Shape Parameter signal detection test for adverse drug reactions." preprint, 2412.05463, https://arxiv.org/abs/2412.05463.

See Also

sim.datagen_tte

pgw *The power generalized Weibull distribution*

Description

Survival, hazard, cumulative distribution, density, quantile and sampling function for the power generalized Weibull (PGW) distribution with parameters `scale`, `shape` and `powershape`.

Usage

```
spgw(x, scale = 1, shape = 1, powershape = 1, log = FALSE)

hpgw(x, scale = 1, shape = 1, powershape = 1, log = FALSE)

ppgw(x, scale = 1, shape = 1, powershape = 1)

dpgw(x, scale = 1, shape = 1, powershape = 1, log = FALSE)

qpgw(p, scale = 1, shape = 1, powershape = 1)

rpgw(n, scale = 1, shape = 1, powershape = 1)
```

Arguments

`x`	vector of quantiles
`scale`	scale parameter
`shape`	shape parameter
`powershape`	power shape parameter
`log`	FALSE (default); if TRUE, the logarithm of the survival probability is returned
`p`	vector of probabilities
`n`	number of observations

Details

The survival function of the PGW distribution is

$$S(x) = \exp\left\{1 - \left[1 + \left(\frac{x}{\theta}\right)^{\nu}\right]^{\frac{1}{\gamma}}\right\}.$$

with scale θ, shape ν and power shape parameter γ.

The hazard function is

$$h(x) = \frac{\nu}{\gamma\theta^{\nu}} \cdot x^{\nu-1} \cdot \left[1 + \left(\frac{x}{\theta}\right)^{\nu}\right]^{\frac{1}{\gamma-1}}$$

The cumulative distribution function is then $F(x) = 1 - S(x)$ and the density function is $f(x) = S(x) \cdot h(x)$. The quantile function is the inverse of the cumulative distribution function $F^{-1}(x)$.

If both shape parameters equal 1, the PGW distribution reduces to the exponential distribution (see dexp) with rate $= 1/$scale If the power shape parameter equals 1, the PGW distribution reduces to the Weibull distribution (see dweibull) with the same parametrization.

If parameter values are not specified, they are set as scale = 1, shape = 1, powershape = 1 per default.

Value

A numeric vector of

- spgw: survival probabilities $S(x)$ (or log-survival if log = TRUE)
- hpgw: hazard values $h(x)$ (or log-hazard if log = TRUE)
- ppgw: cumulative distribution values $F(x)$
- dpgw: density values $f(x)$ (or log-density if log = TRUE)
- qpgw: quantiles corresponding to probabilities p
- rpgw: random samples

References

Nikulin M, Wu HI, others (2016). *The Cox model and its applications*. Springer.

---

plot_pgw *Plot functions of the power generalized Weibull distribution*

Description

Generates plots of the density, cumulative distribution, survival, and hazard functions of the power generalized Weibull (PGW) distribution for specified parameter values.

The function can be used to explore the effect of the scale, shape, and power shape parameters on the distributional form and to support parameter selection for example when specifying prior means for Bayesian modeling.

Usage

```
plot_pgw(scale = 1, shape = 1, powershape = 1)
```

Arguments

scale	scale parameter
shape	shape parameter
powershape	power shape parameter

Value

Produces a four-panel plot showing the density, cumulative distribution, survival, and hazard functions of the PGW distribution.

See Also

pgw

An interactive version of this plot is available on https://janoleko.shinyapps.io/pgwd/.

Examples

```
plot_pgw(scale = 2, shape = 5, powershape = 10)
```

`sim.datagen_tte` *Generate simulated time-to-event data*

Description

Simulation of time-to-event (tte) data.

Usage

```
sim.datagen_tte(genpar)
```

Arguments

`genpar` A vector containing 6 numeric elements:

1. integer sample size N,
2. background rate br in $(0, 1]$ (observed in population on average),
3. adverse drug reaction (ADR) rate adr in $[0, 1]$ as proportion of the background rate,
4. relative proportion of ADR mean time $m.rel$ in $(0, 1)$ of the observation period (OP),
5. relative standard deviation $rel.sd$ in $(0, 1)$ of the OP,
6. length of the OP $censor$.

Details

After specification of the input, the data simulation works as follows:

The absolute number of events due to background causes (other than ADR) is generated by a binomial distribution with probability br. The absolute number of events caused by the ADR is generated with a binomial distribution with probability $br \cdot adr$ giving an expected number of events within the data set is $n \cdot br(1 + adr)$.

For the br cases, the event-times are generated using a uniform distribution on the interval $[0, censor]$. For the ADR cases, event-times are obtained from a normal distribution. The mean of the normal distribution is specified as relative proportion $m.rel$ of the OP (e.g. 0.5 for the middle of the OP). The standard deviation is defined as $rel.sd \cdot censor$. All generated event-times ≤ 0 (due to the normal distribution's support) are set to 1. All generated event-times $\geq censor$ are censored

retroactively. The continuous values are rounded to integer. The data set is filled up with censored observations (status = 0) at time $censor$.

For more details, see Dyck and Sauzet (2025).

Value

A data frame of size N with variables `time` (integer) indicating event- or censoring time and `status` (binary) indicating whether the event was observed or the observation was censored.

References

Dyck J, Sauzet O (2025). "The BPgWSP test: a Bayesian Weibull Shape Parameter signal detection test for adverse drug reactions." preprint, 2412.05463, https://arxiv.org/abs/2412.05463.

See Also

tte

Examples

```
sim.datagen_tte(c(100, 0.1, 1, 0.5, 0.05, 365))
```

`sim.merge_results` *Merge result table batches from simulation study*

Description

Merges result table batches from simulation study obtained from using `sim.run` or `sim.run_parallel` as preparation for evaluation..

Usage

```
sim.merge_results(pc_list, save = TRUE)
```

Arguments

`pc_list`	list of parameter combinations generated with `sim.setup_sim_pars`
`save`	if `TRUE` (default), merged table is saved as res_b.RData or res_f.RData, respectively, in same path where batches are stored; else, result table(s) is/are returned to global environment

Value

Dataframe or list of two dataframes containing all simulation results (one repetition of one simulation scenario per row).

Examples

```
# The package ships with a small precomputed toy simulation study with small
# numbers of repetitions (rep) and posterior sample sizes (stanmod.iter, stanmod.warmup).
# The simulation parameters below match the shipped example results.

#### prep:
toy_path <- system.file("extdata", "toysim", package = "WSPsignal")

# setup prior template
fp_list <- sim.priors_template(tte.dist = c("w", "pgw"), prior.sds = 10)

# fill in prior template with prior means
fp_list$w[,2] <- c(1, 1, 180, 300)
fp_list$w[,3] <- c(1, 0.207, 1, 4)

fp_list$pgw[,2] <- c(1, 1, 20, 300)
fp_list$pgw[,3] <- c(1, 0.207, 5.5, 4)
fp_list$pgw[,4] <- c(1, 1, 14, 1)

# recreate simulation settings used for the toy example
pc_list <- sim.setup_sim_pars(
  N = 500,
  br = 0.1,
  adr.rate = c(0, 1),
  adr.relsd = 0.05,
  study.period = 365,
  est.approach = c("f", "b"),
  tte.dist = c("w", "pgw"),
  prior.dist = "ll",
  fitpars.list = fp_list,
  post.ci.type = c("ETI", "HDI"),
  cred.level = seq(0.5, 0.9, by = 0.05),
  sensitivity.option = 1:3,
  reps = 6,
  batch.size = 3,
  resultpath = toy_path,
  stanmod.iter = 1100,
  stanmod.warmup = 100
)

#### merge Bayesian and frequentist simulation results
res <- sim.merge_results(pc_list, save = FALSE)
res_b <- res$res_b
res_f <- res$res_f
```

`sim.priors_template` *Template for prior specification in simulation study*

Description

Generates a structured template for specifying prior means and prior standard deviations (sds) for the Weibull, double Weibull, or Power generalized Weibull model parameters to be inserted into `sim.setup_sim_pars`.

Usage

```
sim.priors_template(tte.dist = c("w", "dw", "pgw"), prior.sds = 10)
```

Arguments

`tte.dist`	character vector specifying one or multiple modelling approaches; options are `"w"`, `"dw"`, `"pgw"` (see `bwsp_model`)
`prior.sds`	numeric vector setting the same prior sd for all scale and shape parameters across all included model types; default is 10.

Details

The returned list contains one data frame per time-to-event (tte) distribution (`w`, `dw`, `pgw`). For each chosen tte distribution, rows corresponding to different levels of prior belief about the the hazard function are provided, namely `"none"`, `"beginning"`, `"middle"`, and `"end"` (Dyck and Sauzet 2025). Given the template, prior means must be filled by the user before simulation.

Value

A named list containing three data frames `$w`, `$dw`, and `$pgw` specifying prior beliefs and placeholder entries for prior means, and one vector `$prior.sds` specifying the prior standard deviations to be considered per parameter in each prior belief setting.

See Also

`sim.setup_sim_pars`

Examples

```
####  prior elicitation ------------------------------------------------------
# try a few prior parameter combinations and see whether the resulting hazard
# roughly matches the prior belief about the hazard form

# Expected event time can also be taken into account for some guidance, but
# should not be prioritized.
# The reason is that we do not expect the model to accurately fit the hazard of the
# data, but only catch the rough form by distinguishing the cases
# constant vs decreasing vs unimodal vs increasing hazard.

# set prior means for Power generalized Weibull parameters:
plot_pgw(scale = 1, shape = 1, powershape = 1)     # under prior belief "none"
plot_pgw(scale = 20, shape = 5.5, powershape = 14) # under prior belief "beginning"
plot_pgw(scale = 180, shape = 1, powershape = 1)   # under prior belief "middle"
plot_pgw(scale = 300, shape = 4, powershape = 1)   # under prior belief "end"
```

```
#### specify parameter combinations for simulation study -----------------------

fp_list = sim.priors_template(tte.dist = c("pgw"),
                              prior.sds = 10) # setup prior template
# fill in prior template with values chosen in prior elicitation
fp_list$pgw$scale.mean_pgw = c(1, 1, 20, 300)     # scale prior means
fp_list$pgw$shape.mean_pgw <- c(1, 0.207, 5.5, 4) # shape prior means
fp_list$pgw$powershape.mean_pgw = c(1, 1, 14, 1)  # powershape prior means

fp_list # filled fitpars.list ready for sim.setup_sim_pars()
```

`sim.run` *Run simulation*

Description

Runs simulations for all specified data-generating processes, model and test configurations defined in `pc_list`. If simulations for some scenarios have already been completed, the function resumes from the remaining scenarios.

Usage

```
sim.run(pc_list, subset_ind = NULL)
```

Arguments

`pc_list`	list of parameter combinations generated with `sim.setup_sim_pars`
`subset_ind`	vector of integers specifying which rows of `pc_list$pc_table` to be considered in simulation runs; defaults to all rows

Details

This function executes simulations sequentially. For parallel execution, see `sim.run_parallel`.

Value

No return value. This function is called for running simulations and writing results to disk.

Examples

```
## Not run:
# Running simulation studies can take considerable time, especially when
# Bayesian estimation is included. Therefore, this example is not run.

# See ?sim.setup_sim_pars for an example creating pc_list.
```

```
# run all simulation scenarios
sim.run(pc_list) # run all simulation scenarios

# to run only a subset of simulation scenarios, specify row indices from
# pc_list$pc_table, e.g. the first 10 scenarios:
sim.run(pc_list, subset_ind = 1:10)

## End(Not run)
```

`sim.run_parallel` *Run simulation (in parallel)*

Description

Runs simulations in parallel for all specified data-generating processes, model and test configurations defined in `pc_list`. If simulations for some scenarios have already been completed, the function resumes from the remaining scenarios.

Usage

```
sim.run_parallel(pc_list, subset_ind = NULL)
```

Arguments

`pc_list`	list of parameter combinations generated with `sim.setup_sim_pars`
`subset_ind`	vector of integers specifying which rows of `pc_list$pc_table` to be considered in simulation runs; defaults to all rows

Details

Parallelization needs to be initialized using the `plan` command (see example).

Value

No return value. This function is called for running simulations and writing results to disk.

Examples

```
## Not run:
# Running simulation studies can take considerable time, especially when
# Bayesian estimation is included. Therefore, this example is not run.

# See ?sim.setup_sim_pars for an example creating pc_list.

# prep parallelization
# install.packages("future")
future::plan(future::multisession,
```

```
              workers = future::availableCores())

# run all simulation scenarios
sim.run_parallel(pc_list)

# to run only a subset of simulation scenarios, specify row indices from
# pc_list$pc_table, e.g. the first 10 scenarios:
sim.run_parallel(pc_list, subset_ind = 1:10)

## End(Not run)
```

`sim.setup_sim_pars` *Set up simulation parameters*

Description

Sets up parameters for a simulation study to tune the Weibull shape parameter (WSP) test. Simulation parameters encompass data generating process (DGP) parameters (`N`, `...`, `study.period`), tuning parameters for the WSP test (`est.approach`, `...`, `sensitivity.option`), and additional parameters (`reps`, `...`, `stanmod.warmup`).

Usage

```
sim.setup_sim_pars(
  N,
  br,
  adr.rate,
  adr.relsd,
  study.period,
  est.approach,
  tte.dist,
  prior.dist,
  fitpars.list,
  post.ci.type = c("ETI", "HDI"),
  cred.level,
  sensitivity.option = 1:3,
  reps = 100,
  batch.size = 10,
  resultpath,
  stanmod.chains = 4,
  stanmod.iter = 11000,
  stanmod.warmup = 1000
)
```

Arguments

N	vector of sample sizes
br	vector of background rates (observed in population on average)
adr.rate	vector of adverse drug reaction rates as proportions of the background rates
adr.relsd	vector of relative standard deviations from the adverse drug reaction times
study.period	scalar specifying the length of the study period
est.approach	character vector specifying one or two estimation approaches; options are Bayesian "b" and frequentist "f"
tte.dist	character vector specifying one or multiple modelling approaches; options are "w", "dw", "pgw" (see bwsp_model, fwsp_model)
prior.dist	character indicating the prior distribution for the parameters of the Bayesian time-to-event (tte) distribution; options are "fg", "fl", "gg", "ll" (see bwsp_model)
fitpars.list	list with one dataframe per tte distribution containing the prior specifications for BWSP model fitting; setup with sim.priors_template
post.ci.type	character indicating whether to extract equal tailed intervals ("ETI") or highest posterior density intervals ("HDI") as credibility interval (CI) for BWSP testing (see bwsp_test)
cred.level	vector of credibility levels used for construction of region of practical equivalence (ROPE) and posterior CI for the BWSP tests (see bwsp_test) or vector of confidence levels for FWSP tests (see fwsp_test).
sensitivity.option	vector of sensitivity options for the BWSP test (see bwsp_test)
reps	number of repetitions for each simulation scenario, default is 100
batch.size	number of simulation repetitions to be saved in a batch (see details); default is 10
resultpath	directory where output of this function and intermediate results of the simulation are stored; needs explicit specification
stanmod.chains	number of Markov chains (see sampling); default is 4
stanmod.iter	total number of iterations per chain including warmup (see sampling); default is 11000
stanmod.warmup	number of warmup (aka burn-in) iterations per chain (see sampling); default is 1000

Details

The purpose of the simulation study is to evaluate the performance of different WSP tests for data scenarios of interest following the tuning scheme developed in Dyck and Sauzet (2025).

DGP parameters (N, ..., study.period) should reflect the data characteristics of interest. Within simulation, data are generated with sim.datagen_tte.

Tuning parameters for the WSP test (est.approach, ..., sensitivity.option) lead to a range of tuning combinations for the WSP tests evaluated during the simulation study to find the best test specification. Argument fitpars.list needs to be prepared with sim.priors_template. Note, that the tte.dist argument in sim.priors_template and in sim.setup_sim_pars must match.

Additional parameters (`reps, ..., stanmod.warmup`) specify simulation settings and specifications for posterior sampling. Simulation settings encompass the number of repetitions per simulation scenario, the directory in which to save results and batch saving. Batch saving is done to prevent losing simulation results in case of an interruption of simulation e.g. due to termination of the R session. Posterior sampling specifications encompass the number of chains, iterations and the length of the warmup phase. See `sampling` for more details on the posterior estimation function.

Value

A list containing all simulation specifications in the format required for `sim.run` or `sim.run_parallel`.

References

Dyck J, Sauzet O (2025). "The BPgWSP test: a Bayesian Weibull Shape Parameter signal detection test for adverse drug reactions." preprint, 2412.05463, https://arxiv.org/abs/2412.05463.

Examples

```
#### specify parameter combinations for simulation study

# setup prior template
fp_list = sim.priors_template(tte.dist = c("w", "pgw"), prior.sds = 10)
fp_list # fitpars.list template

# fill in prior template with values chosen in prior elicitation
# for weibull models:
fp_list$w[,2] = c(1, 1, 180, 300) # scale prior means
fp_list$w[,3] = c(1, 0.207, 1, 4) # shape prior means

# for pgw models:
fp_list$pgw[,2] = c(1, 1, 20, 300)   # scale prior means
fp_list$pgw[,3] = c(1, 0.207, 5.5, 4)# shape prior means
fp_list$pgw[,4] = c(1, 1, 14, 1)     # powershape prior means

fp_list # fitpars.list filled with means

# setup parameter combination list for simulation
pc_list = sim.setup_sim_pars(N = c(500, 3000, 5000),            # DGP parameters
                             br = 0.1,                          # |
                             adr.rate = c(0, 0.5, 1),           # |
                             adr.relsd = 0.05,                  # |
                             study.period = 365,                # v
                             est.approach = c("f", "b"),        # Tuning parameters
                             tte.dist = c("w", "pgw"),          # |
                             prior.dist = c("ll", "gg"),        # |
                             fitpars.list = fp_list,            # |
                             post.ci.type = c("ETI", "HDI"),    # |
                             cred.level = c(seq(0.5,0.9, by = 0.05)),
                             sensitivity.option = 1:3,          # v
                             reps = 100,                        # Additional parameters
                             batch.size = 10,                   # |
                             resultpath = tempdir(),            # |
```

```
                              stanmod.iter = 11000,              # |
                              stanmod.warmup = 1000              # v
                              )
```

```
pc_list
```

`tte` *test time-to-event dataset*

Description

Simulated time-to-event (tte) data generated with `sim.datagen_tte(c(100, 0.1, 1, 0.5, 0.05, 365))`.

Usage

```
tte
```

Format

A data frame with 100 rows and 2 variables:

time event time if an event was observed, or censoring time if no event was observed,

status event status; 1 if an event was observed, 0 if no event was observed.

See Also

`sim.datagen_tte`

`tte2priordat` *Prior and data setup for Bayesian survival model fitting*

Description

Prepare time-to-event data and prior specifications for model fitting with `bwsp_model`.

Usage

```
tte2priordat(
  dat,
  tte.dist = "dw",
  prior.dist = "ll",
  scale.mean = 1,
  scale.sd = 10,
  shape.mean = 1,
  shape.sd = 10,
  scale_c.mean = 1,
  scale_c.sd = 10,
  shape_c.mean = 1,
  shape_c.sd = 10,
  powershape.mean = 1,
  powershape.sd = 10
)
```

Arguments

dat	matrix or data frame with time in the first column and event status in the second column
tte.dist	character indicating the modelling approach; options are "w", "dw", "pgw"; default is "dw"
prior.dist	character indicating the prior distribution for the parameters of the tte distribution; options are "fg", "fl", "gg", "ll"; default is "ll"
scale.mean	prior mean of the scale parameter; default is 1
scale.sd	prior standard deviation (sd) of the scale parameter; default is 10
shape.mean	prior mean of the shape parameter; default is 1
shape.sd	prior sd of the shape parameter; default is 10
scale_c.mean	prior mean of the scale parameter for censored-at-half data (only for tte.dist="dw"); default is 1
scale_c.sd	prior sd of the scale parameter for censored-at-half data (only for tte.dist="dw"); default is 10
shape_c.mean	prior mean of the shape parameter for censored-at-half data (only for tte.dist="dw"); default is 1
shape_c.sd	prior sd of the shape parameter for censored-at-half data (only for tte.dist="dw"); default is 10
powershape.mean	prior mean of the power shape parameter (only for tte.dist="pgw")
powershape.sd	prior sd of the power shape parameter (only for tte.dist="pgw")

Details

The function prepares data to fit a Bayesian model to time-to-event data. The distribution can be assumed a Weibull ("w"), a double Weibull ("dw", estimating two Weibull models - one to the data

as is and one to the data censored at mid of observation period), or a power generalized Weibull (`"pgw"`) model.

Only the parameters relevant to the chosen `tte.dist` need to be provided, that is:

- for `"w"`: `scale.mean`, `scale.sd`, `shape.mean`, `shape.sd`
- for `"dw"`: `scale.mean`, `scale.sd`, `shape.mean`, `shape.sd`, `scale_c.mean`, `scale_c.sd`, `shape_c.mean`, `shape_c.sd`
- for `"pgw"`: `scale.mean`, `scale.sd`, `shape.mean`, `shape.sd`, `powershape.mean`, `powershape.sd`

Implemented prior distributions for the scale and shape parameters are products of the following univariate distributional choices:

for scale parameter	for shape parameter(s)	abbreviation
fixed to prior mean	gamma	fg
gamma	gamma	gg
fixed to prior mean	lognormal	fl
lognormal	lognormal	ll

Prior means suitable to reflect the prior belief can be worked out by plotting the hazard and estimating the expected event time under different parameter combinations using `plot_pgw` (setting argument `powershape = 1` reduces the power generalized Weibull distribution to Weibull) or `https://janoleko.shinyapps.io/pgwd/`. The default values for prior means correspond to the prior belief of constant hazard.

Prior standard deviations should reflect the uncertainty about the prior belief (i.e. set smaller standard deviation in case of high certainty about prior belief vs. larger standard deviation in case of low certainty).

Value

A named list in the format expected by `bwsp_model`.

Examples

```
tte2priordat(dat = tte, tte.dist = "w", prior.dist = "ll",
             scale.mean = 10, scale.sd = 2,
             shape.mean = 1.5, shape.sd = 15)

tte2priordat(dat = tte, tte.dist = "dw", prior.dist = "ll",
             scale.mean = 10, scale.sd = 2,
             shape.mean = 1.5, shape.sd = 15,
             scale_c.mean = 5, scale_c.sd = 1,
             shape_c.mean = 1, shape_c.sd = 10)

tte2priordat(dat = tte, tte.dist = "pgw", prior.dist = "ll",
             scale.mean = 10, scale.sd = 2,
             shape.mean = 1.5, shape.sd = 15,
             powershape.mean = 3, powershape.sd = 20)
```

Index