tcplFit2

January 6, 2020

Type Package

Title Performs Concentration-Response Modeling on HTS or transcriptomics data

Version 0.1.0

Author Thomas Sheffield

Maintainer Richard Judson <judson.richard@epa.gov>

Description This package performs the basic concentration response curve fitting used in the tcpl package. It is a
substitute for the original tcplFit function (and sub-functions) and allows a wider variety of concentration-response models. All of the models included in the BMDExpress package are now part of this package,
and the output includes a calculation of the bmd (Benchmark Dse or concentration) value

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.0.2

Suggests knitr,
rmarkdown

VignetteBuilder knitr

Imports stats,
RColorBrewer

R topics documented:

`acgnlsobj` *AC GNLS Objective Function*

Description

GNLS objective function set to y for gnls solver.

Usage

```
acgnlsobj(x, y, tp, ga, p, la, q)
```

Arguments

x	Concentration.
y	Desired activity level.
tp	Top.
ga	Gain AC50.
p	Gain power.
la	Loss AC50.
q	Loss power.

Value

Difference between GNLS model repsone at x and y.

acy *Activity Concentration y*

Description

Returns concentration at which model equals y.

Usage

```
acy(
  y,
  modpars,
  type = "hill",
  returntop = F,
  returntoploc = F,
  getloss = F,
  verbose = F
)
```

Arguments

y	Activity value at which the concentration is desired. y should be less than the model's top, if there is one, and greater than zero.
modpars	List of named model parameters. Model parameters can include: "a", "b", "ga", "la", "p", "q", "tp". ga and la should NOT be in log units.
type	Model type; must be one of: "exp1", "exp2", "exp3", "exp4", "gnls", "hill", "poly1", "poly2", "pow".
returntop	When TRUE, returns actual top value for gnls. Has no effect for other models.
returntoploc	When TRUE, returns concentration of top for gnls. Has no effect for other models. If top location can't be found, NA is returned.
getloss	When TRUE, returns value on loss side of curve for gnls. Has no effect for other models.
verbose	When TRUE, shows warnings.

Details

Mathematically inverts model functions of the given type, except for gnls, which is numerically inverted. gnls returns NA when y > tp. Other options return the actual top (as opposed to theoretical tp) and top location for gnls model. gnls model defaults to giving concentration on gain side. Only one of getloss, returntop, and returntoploc should be TRUE at a time. If top location solution fails for gnls, top is set to tp. Returns NA if gnls numerical solver fails.

Value

Ouputs concentration at activity y, or gnls top or top concentration, when applicable.

Examples

```
acy(1, list(ga = 10, tp = 2, p = 3), type = "hill")
acy(1, list(ga = .1, tp = 2, p = 3, q = 3,la = 10), type = "gnls")
acy(1, list(ga = .1, tp = 2, p = 3, q = 3,la = 10), type = "gnls", getloss = TRUE)
acy(1, list(ga = .1, tp = 2, p = 3, q = 3,la = 10), type = "gnls", returntop = TRUE)
acy(1, list(ga = .1, tp = 2, p = 3, q = 3,la = 10), type = "gnls", returntoploc = TRUE)
```

bmdbounds *BMD Bounds*

Description

Computes BMDU or BMDL.

Usage

```
bmdbounds(
  fit_method,
  bmr,
  pars,
  conc,
  resp,
  onesidedp = 0.05,
  bmd = NULL,
  which.bound = "lower"
)
```

Arguments

fit_method	Fit method: "exp2", "exp3", "exp4", "exp5", "hill", "gnls", "poly1", "poly2", or "pow".
bmr	Benchmark response.
pars	Named vector of model parameters: a,b,tp,ga,p,la,q,er output by httrfit, and in that order.

`conc` Vector of concentrations (NOT in log units).

`resp` Vector of responses corresponding to given concentrations.

`onesidedp` The one-sided p-value. Default of .05 corresponds to 5 percentile BMDL, 95 percentile BMDU, and 90 percent CI.

`bmd` Can optionally input the bmd when already known to avoid unnecessary calculation.

`which.bound` Returns BMDU if which.bound = "upper"; returns BMDL if which.bound = "lower".

Details

Takes in concentration response fit details and outputs a bmdu or bmdl, as desired. If bmd is not finite, returns NA. If the objective function doesn't change sign or the root finding otherwise fails, it returns NA. These failures are not uncommon since some curves just don't reach the desired confidence level.

Value

Returns either the BMDU or BMDL.

Examples

```
conc = c(.03, .1, .3, 1, 3, 10, 30, 100)
resp = c(.1,-.1,0,1.1,1.9,2,2.1,1.9)
pars = c(tp = 1.973356, ga = 0.9401224, p = 3.589397, er =  -2.698579)
bmdbounds(fit_method = "hill", bmr = .5, pars, conc, resp)
bmdbounds(fit_method = "hill", bmr = .5, pars, conc, resp, which.bound = "upper")
```

`bmdobj` *BMD Objective Function*

Description

Utility function for bmdbounds

Usage

```
bmdobj(bmd, fname, bmr, conc, resp, ps, mll, onesp, partype = 2)
```

Arguments

`bmd` Benchmark dose.

`fname` Function name: "exp2", "exp3", "exp4", "exp5", "hillfn", "gnls", "poly1", "poly2", or "pow".

`bmr` Benchmark response.

`conc` Vector of concentrations NOT in log units.

resp Vector of corresponding responses.

ps Named list of paramters.

mll Maximum log-likelihood of winning model.

onesp One-sided p-value.

partype Number for parameter type. Type 1 is y-scaling: a or tp. Type 2 is x-scaling: b or ga, when available, a otherwise. Type 3 is power scaling: p when available, then b or ga, then a if no others. Since bmd is linked to the x-scale, type 2 should always be used. Other types can also be vulnerable to underflow/overflow.

Value

Objective function value to find the zero of.

cnst *Constant Model*

Description

Constant Model

Usage

```
cnst(ps, x)
```

Arguments

ps Vector of parameters (ignored)

x Vector of concentrations (regular units)

Value

Vector of model responses

concRespCore *Concentration Response Core*

Description

Core of concentration response curve fitting for pvalue based cutoff. This function calls httrFit to get curve fits, chooses the winning model, extracts the top and ac50, computes the hitcall, and calculates bmd/bmdl/bmdu among other statistics. Nested model selection is used to choose between poly1/poly2, then the model with the lowest AIC (or AICc) is declared the winner. Continuous hitcalls requires tcplFit2 to be run with force.fit = T and "cnst" never to be chosen as the winner.

Usage

```
concRespCore(
  row,
  fitmodels = c("cnst", "hill", "gnls", "poly1", "poly2", "pow", "exp2", "exp3", "exp4",
    "exp5"),
  conthits = T,
  aicc = F,
  force.fit = FALSE,
  bidirectional = TRUE,
  verbose = FALSE,
  do.plot = F
)
```

Arguments

`row` A named list that must include:

- conc - list of concentrations (not in log units)
- resp - list of corresponding responses
- bmed - median of noise estimate.
- cutoff - noise cutoff
- onesd - 1 standard deviation of the noise (for bmd calculation)

Other elements (usually identifiers, like casrn) of row will be attached to the final output.

`fitmodels` Vector of model names to use.

`conthits` conthits = T uses continuous hitcalls, otherwise they're discrete.

`aicc` aicc = T uses corrected AIC to choose winning method; otherwise regular AIC.

`force.fit` If TRUE force the fitting to proceed even if there are no points outside of the bounds (default FALSE)

`bidirectional` If TRUE allow fitting to happen in both directions (default TRUE)

`verbose` If TRUE, write extra output from tcplFit2 (default FALSE)

`do.plot` If TRUE, create a plot in the tcplFit2 function (default FALSE)

Value

One row dataframe containing all CR output and statistics and any identifiers from row.

Examples

```
conc = list(.03,.1,.3,1,3,10,30,100)
resp = list(0,.2,.1,.4,.7,.9,.6, 1.2)
row = list(conc = conc, resp = resp, bmed = 0, cutoff = 1, onesd = .5)
concRespCore(row, conthits = TRUE)
concRespCorec(row, aicc = TRUE)
```

exp2 *Exponential 2 Model*

Description

Exponential 2 Model

Usage

```
exp2(ps, x)
```

Arguments

ps	Vector of parameters: a,b,er
x	Vector of concentrations (regular units)

Value

Vector of model responses

exp3 *Exponential 3 Model*

Description

Exponential 3 Model

Usage

```
exp3(ps, x)
```

Arguments

ps	Vector of parameters: a,b,p,er
x	Vector of concentrations (regular units)

Value

Vector of model responses

exp4 *Exponential 4 Model*

Description

Exponential 4 Model

Usage

```
exp4(ps, x)
```

Arguments

`ps`	Vector of parameters: tp,ga,er
`x`	Vector of concentrations (regular units)

Value

Vector of model responses

exp5 *Exponential 5 Model*

Description

Exponential 5 Model

Usage

```
exp5(ps, x)
```

Arguments

`ps`	Vector of parameters: tp,ga,p,er
`x`	Vector of concentrations (regular units)

Value

Vector of model responses

fitcnst *Constant Model Fit*

Description

Function that fits a constant line and returns generic model outputs.

Usage

```
fitcnst(conc, resp, nofit = F)
```

Arguments

conc	Vector of concentration values NOT in log units.
resp	Vector of corresponding responses.
nofit	If nofit = T, returns formatted output filled with missing values.

Details

success = 1 for a successful fit, 0 if optimization failed, and NA if nofit = T. aic, rme, and er are set to NA in case of nofit or failure. pars always equals "er".

Value

List of five elements: success, aic (Aikaike Information Criteria), rme (root mean square error), er (error parameter), pars (parameter names).

Examples

```
fitcnst(c(.1,1,10,100), c(1,2,0,-1))
fitcnst(c(.1,1,10,100), c(1,2,0,-1), nofit = TRUE)
```

fitexp2 *Exponential 2 Model Fit*

Description

Function that fits to f(x) = a*(e^(x/b)- 1) and returns generic model outputs.

Usage

```
fitexp2(conc, resp, bidirectional = TRUE, verbose = FALSE, nofit = F)
```

Arguments

`conc`	Vector of concentration values NOT in log units.
`resp`	Vector of corresponding responses.
`bidirectional`	If TRUE, model can be positive or negative; if FALSE, it will be positive only.
`verbose`	If TRUE, gives optimization and hessian inversion details.
`nofit`	If nofit = T, returns formatted output filled with missing values.

Details

Zero background and increasing absolute response are assumed. Parameters are "a" (y scale), "b" (x scale), and error term "er". success = 1 for a successful fit, 0 if optimization failed, and NA if nofit = T. cov = 1 for a successful hessian inversion, 0 if it fails, and NA if nofit = T. aic, rme, modl, parameters, and parameter sds are set to NA in case of nofit or failure.

Value

Named list containing: success, aic (Aikaike Information Criteria), cov (success of covariance calculation), rme (root mean square error), modl (vector of model values at given concentrations), parameters values, parameter sd (standard deviation) estimates, pars (vector of parameter names), sds (vector of parameter sd names).

Examples

```
fitexp2(c(.1,1,10,100), c(0,.1,1,10))
```

`fitexp3` *Exponential 3 Model Fit*

Description

Function that fits to f(x) = a*(e^((x/b)^p)- 1) and returns generic model outputs.

Usage

```
fitexp3(
  conc,
  resp,
  bidirectional = TRUE,
  verbose = FALSE,
  nofit = F,
  dmin = 0.3
)
```

Arguments

`conc` Vector of concentration values NOT in log units.

`resp` Vector of corresponding responses.

`bidirectional` If TRUE, model can be positive or negative; if FALSE, it will be positive only.

`verbose` If TRUE, gives optimization and hessian inversion details.

`nofit` If nofit = T, returns formatted output filled with missing values.

`dmin` Minimum allowed value of p.

Details

Zero background and increasing absolute response are assumed. Parameters are "a" (y scale), "b" (x scale), "p" (power), and error term "er". success = 1 for a successful fit, 0 if optimization failed, and NA if nofit = T. cov = 1 for a successful hessian inversion, 0 if it fails, and NA if nofit = T. aic, rme, modl, parameters, and parameter sds are set to NA in case of nofit or failure.

Value

Named list containing: success, aic (Aikaike Information Criteria), cov (success of covariance calculation), rme (root mean square error), modl (vector of model values at given concentrations), parameters values, parameter sd (standard deviation) estimates, pars (vector of parameter names), sds (vector of parameter sd names).

Examples

```
fitexp3(c(.03,.1,.3,1,3,10,30,100), c(0,0,.1, .2, .4, 1, 4, 50))
```

fitexp4 *Exponential 4 Model Fit*

Description

Function that fits to f(x) = tp*(1-2^(-x/ga)) and returns generic model outputs.

Usage

```
fitexp4(conc, resp, bidirectional = TRUE, verbose = FALSE, nofit = F)
```

Arguments

`conc` Vector of concentration values NOT in log units.

`resp` Vector of corresponding responses.

`bidirectional` If TRUE, model can be positive or negative; if FALSE, it will be positive only.

`verbose` If TRUE, gives optimization and hessian inversion details.

`nofit` If nofit = T, returns formatted output filled with missing values.

Details

Zero background and increasing absolute response are assumed. Parameters are "tp" (top), "ga" (AC50), and error term "er". success = 1 for a successful fit, 0 if optimization failed, and NA if nofit = T. cov = 1 for a successful hessian inversion, 0 if it fails, and NA if nofit = T. aic, rme, modl, parameters, and parameter sds are set to NA in case of nofit or failure.

Value

Named list containing: success, aic (Aikaike Information Criteria), cov (success of covariance calculation), rme (root mean square error), modl (vector of model values at given concentrations), parameters values, parameter sd (standard deviation) estimates, pars (vector of parameter names), sds (vector of parameter sd names).

Examples

```
fitexp4(c(.03,.1,.3,1,3,10,30,100), c(0,0,.1, .2, .5, 1, 1.5, 2))
```

`fitexp5` *Exponential 5 Model Fit*

Description

Function that fits to f(x) = tp*(1-2^(-(x/ga)^p)) and returns generic model outputs.

Usage

```
fitexp5(
  conc,
  resp,
  bidirectional = TRUE,
  verbose = FALSE,
  nofit = F,
  dmin = 0.3
)
```

Arguments

`conc`	Vector of concentration values NOT in log units.
`resp`	Vector of corresponding responses.
`bidirectional`	If TRUE, model can be positive or negative; if FALSE, it will be positive only.
`verbose`	If TRUE, gives optimization and hessian inversion details.
`nofit`	If nofit = T, returns formatted output filled with missing values.
`dmin`	Minimum allowed value of p.

Details

Zero background and increasing absolute response are assumed. Parameters are "tp" (top), "ga" (AC50), "p" (power), and error term "er". success = 1 for a successful fit, 0 if optimization failed, and NA if nofit = T. cov = 1 for a successful hessian inversion, 0 if it fails, and NA if nofit = T. aic, rme, modl, parameters, and parameter sds are set to NA in case of nofit or failure.

Value

Named list containing: success, aic (Aikaike Information Criteria), cov (success of covariance calculation), rme (root mean square error), modl (vector of model values at given concentrations), parameters values, parameter sd (standard deviation) estimates, pars (vector of parameter names), sds (vector of parameter sd names).

Examples

```
fitexp5(c(.03,.1,.3,1,3,10,30,100), c(0,0,.1, .2, .5, 1, 1.5, 2))
```

`fitgnls` *Gain-Loss Model Fit*

Description

Function that fits to f(x) = tp/[(1 + (ga/x)^p)(1 + (x/la)^q)] and returns generic model outputs.

Usage

```
fitgnls(
  conc,
  resp,
  bidirectional = TRUE,
  verbose = FALSE,
  nofit = F,
  minwidth = 1.5
)
```

Arguments

`conc`	Vector of concentration values NOT in log units.
`resp`	Vector of corresponding responses.
`bidirectional`	If TRUE, model can be positive or negative; if FALSE, it will be positive only.
`verbose`	If TRUE, gives optimization and hessian inversion details.
`nofit`	If nofit = T, returns formatted output filled with missing values.
`minwidth`	Minimum allowed distance between gain ac50 and loss ac50 (in log10 units).

Details

Concentrations are converted internally to log10 units and optimized with f(x) = tp/[(1 + 10^(p*(ga-x)))(1 + 10^(q*(x-la)))], then ga, la, ga_sd, and la_sd are converted back to regular units before returning. Zero background and increasing initial absolute response are assumed. Parameters are "tp" (top), "ga" (gain AC50), "p" (gain power), "la" (loss AC50),"q" (loss power) and error term "er". success = 1 for a successful fit, 0 if optimization failed, and NA if nofit = T. cov = 1 for a successful hessian inversion, 0 if it fails, and NA if nofit = T. aic, rme, modl, parameters, and parameter sds are set to NA in case of nofit or failure.

Value

Named list containing: success, aic (Aikaike Information Criteria), cov (success of covariance calculation), rme (root mean square error), modl (vector of model values at given concentrations), parameters values, parameter sd (standard deviation) estimates, pars (vector of parameter names), sds (vector of parameter sd names).

Examples

```
fitgnls(c(.03,.1,.3,1,3,10,30,100), c(0,.3,1, 2, 2.1, 1.5, .8, .2))
```

`fithill` *Hill Model Fit*

Description

Function that fits to f(x) = tp/[(1 + (ga/x)^p)] and returns generic model outputs.

Usage

```
fithill(conc, resp, bidirectional = TRUE, verbose = FALSE, nofit = F)
```

Arguments

`conc`	Vector of concentration values NOT in log units.
`resp`	Vector of corresponding responses.
`bidirectional`	If TRUE, model can be positive or negative; if FALSE, it will be positive only.
`verbose`	If TRUE, gives optimization and hessian inversion details.
`nofit`	If nofit = T, returns formatted output filled with missing values.

Details

Concentrations are converted internally to log10 units and optimized with f(x) = tp/(1 + 10^(p*(ga-x))), then ga and ga_sd are converted back to regular units before returning. Zero background and increasing initial absolute response are assumed. Parameters are "tp" (top), "ga" (gain AC50), "p" (gain power), and error term "er". success = 1 for a successful fit, 0 if optimization failed, and NA if nofit = T. cov = 1 for a successful hessian inversion, 0 if it fails, and NA if nofit = T. aic, rme, modl, parameters, and parameter sds are set to NA in case of nofit or failure.

Value

Named list containing: success, aic (Aikaike Information Criteria), cov (success of covariance calculation), rme (root mean square error), modl (vector of model values at given concentrations), parameters values, parameter sd (standard deviation) estimates, pars (vector of parameter names), sds (vector of parameter sd names).

Examples

```
fithill(c(.03,.1,.3,1,3,10,30,100), c(0,0,.1, .2, .5, 1, 1.5, 2))
```

`fitpoly1` *Polynomial 1 (Linear) Model Fit*

Description

Function that fits to f(x) = a*x and returns generic model outputs.

Usage

```
fitpoly1(conc, resp, bidirectional = TRUE, verbose = FALSE, nofit = F)
```

Arguments

`conc`	Vector of concentration values NOT in log units.
`resp`	Vector of corresponding responses.
`bidirectional`	If TRUE, model can be positive or negative; if FALSE, it will be positive only.
`verbose`	If TRUE, gives optimization and hessian inversion details.
`nofit`	If nofit = T, returns formatted output filled with missing values.

Details

Zero background and increasing absolute response are assumed. Parameters are "a" (y scale) and error term "er". success = 1 for a successful fit, 0 if optimization failed, and NA if nofit = T. cov = 1 for a successful hessian inversion, 0 if it fails, and NA if nofit = T. aic, rme, modl, parameters, and parameter sds are set to NA in case of nofit or failure.

Value

Named list containing: success, aic (Aikaike Information Criteria), cov (success of covariance calculation), rme (root mean square error), modl (vector of model values at given concentrations), parameters values, parameter sd (standard deviation) estimates, pars (vector of parameter names), sds (vector of parameter sd names).

Examples

```
fitpoly1(c(.03,.1,.3,1,3,10,30,100), c(0,.01,.1, .1, .2, .5, 2, 5))
```

fitpoly2 *Polynomial 2 (Quadratic) Model Fit*

Description

Function that fits to f(x) = a*(x/b + x^2/b^2) and returns generic model outputs.

Usage

```
fitpoly2(conc, resp, bidirectional = TRUE, verbose = FALSE, nofit = F)
```

Arguments

`conc`	Vector of concentration values NOT in log units.
`resp`	Vector of corresponding responses.
`bidirectional`	If TRUE, model can be positive or negative; if FALSE, it will be positive only.
`verbose`	If TRUE, gives optimization and hessian inversion details.
`nofit`	If nofit = T, returns formatted output filled with missing values.

Details

Zero background and monotonically increasing absolute response are assumed. Parameters are "a" (y scale), "b" (x scale), and error term "er". success = 1 for a successful fit, 0 if optimization failed, and NA if nofit = T. cov = 1 for a successful hessian inversion, 0 if it fails, and NA if nofit = T. aic, rme, modl, parameters, and parameter sds are set to NA in case of nofit or failure.

Value

Named list containing: success, aic (Aikaike Information Criteria), cov (success of covariance calculation), rme (root mean square error), modl (vector of model values at given concentrations), parameters values, parameter sd (standard deviation) estimates, pars (vector of parameter names), sds (vector of parameter sd names).

Examples

```
fitpoly2(c(.03,.1,.3,1,3,10,30,100), c(0,.01,.1, .1, .2, .5, 2, 8))
```

`fitpow` *Power Model Fit*

Description

Function that fits tof(x) = a*x^p and returns generic model outputs.

Usage

```
fitpow(
  conc,
  resp,
  bidirectional = TRUE,
  verbose = FALSE,
  nofit = F,
  nmin = 0.3
)
```

Arguments

`conc`	Vector of concentration values NOT in log units.
`resp`	Vector of corresponding responses.
`bidirectional`	If TRUE, model can be positive or negative; if FALSE, it will be positive only.
`verbose`	If TRUE, gives optimization and hessian inversion details.
`nofit`	If nofit = T, returns formatted output filled with missing values.
`nmin`	Minimum allowed value of p.

Details

Zero background and monotonically increasing absolute response are assumed. Parameters are "a" (y scale), "p" (power), and error term "er". success = 1 for a successful fit, 0 if optimization failed, and NA if nofit = T. cov = 1 for a successful hessian inversion, 0 if it fails, and NA if nofit = T. aic, rme, modl, parameters, and parameter sds are set to NA in case of nofit or failure.

Value

Named list containing: success, aic (Aikaike Information Criteria), cov (success of covariance calculation), rme (root mean square error), modl (vector of model values at given concentrations), parameters values, parameter sd (standard deviation) estimates, pars (vector of parameter names), sds (vector of parameter sd names).

Examples

```
fitpow(c(.03,.1,.3,1,3,10,30,100), c(0,.01,.1, .1, .2, .5, 2, 8))
```

gnls *Gain-Loss Model*

Description

Gain-Loss Model

Usage

```
gnls(ps, x)
```

Arguments

`ps`	Vector of parameters: tp,ga,p,la,q,er
`x`	Vector of concentrations (regular units)

Value

Vector of model responses

gnlsderivobj *GNLS Derivative Objective Function*

Description

Derivative of the gnls function set to zero for top location solver.

Usage

```
gnlsderivobj(x, tp, ga, p, la, q)
```

Arguments

`x`	Concentration.
`tp`	Top.
`ga`	Gain AC50.
`p`	Gain power.
`la`	Loss AC50.
`q`	Loss power.

Value

Value of gnls derivative at x.

hello *Hello, World!*

Description

Prints 'Hello, world!'.

Usage

```
hello()
```

Examples

```
hello()
```

hillfn *Hill Model*

Description

Hill Model

Usage

```
hillfn(ps, x)
```

Arguments

ps	Vector of parameters: tp,ga,p,er
x	Vector of concentrations (regular units)

Value

Vector of model responses

hitcont *Continuous Hitcalls*

Description

Wrapper that computes continuous hitcalls for a provided PATHWAY_CR dataframe.

Usage

```
hitcont(indf, xs = NULL, ys = NULL, newcutoff, mc.cores = 1)
```

Arguments

`indf`	Dataframe similar to PATHWAY_CR. Must contain "conc" and "resp" columns if xs and ys are not provided. Must contain "top", "ac50", "er", "fit_method", "caikwt", and "mll" columns as well as columns for each model parameter.
`xs`	List of concentration vectors that can be provided for speed.
`ys`	List of response vectors that can be provided for speed.
`newcutoff`	Vector of new cutoff values to use. Length should be equal to rows in indf.
`mc.cores`	Number of cores to use for large dataframes.

Details

indf parameter columns should be NA when not required by fit method. "conc" and "resp" entries should be a single string with values separated by |. Details on indf columns can be found in pathwayConcRespCore_pval.

Value

Vector of hitcalls between 0 and 1 with length equal to indf row number.

hitcontinner *Continuous Hitcalls Inner*

Description

Calculates continuous hitcall using 3 statistical metrics.

Usage

```
hitcontinner(conc, resp, top, cutoff, er, ps, fit_method, caikwt, mll)
```

Arguments

`conc`	Vector of concentrations.
`resp`	Vector of responses.
`top`	Model top.
`cutoff`	Desired cutoff.
`er`	Model error parameter.
`ps`	Vector of used model parameters in order: a, tp, b, ga, p, la, q, er.
`fit_method`	Name of winning fit method (should never be constant).
`caikwt`	Aikaike weight of constant model relative to winning model.
`mll`	Maximum log-likelihood of winning model.

Details

This function is called either directly from pathwayConcRespCore_pval or via hitcont using PATHWAY_CR. Details of how to compute function input are in pathwayConcRespCore_pval.

Value

Continuous hitcall between 0 and 1.

Examples

```
conc = c(.03,.1,.3,1,3,10,30,100)
resp = c(0,.1,0,.2,.6,.9,1.1,1)
top = 1.023239
er = -3.295307
ps = c(1.033239, 2.453014, 1.592714, er = -3.295307) #tp,ga,p,er
fit_method = "hill"
caikwt = 1.446966e-08
mll = 12.71495
hitcontinner(conc,resp,top,cutoff = 0.8, er,ps,fit_method, caikwt, mll)
hitcontinner(conc,resp,top,cutoff = 1, er,ps,fit_method, caikwt, mll)
hitcontinner(conc,resp,top,cutoff = 1.2, er,ps,fit_method, caikwt, mll)
```

`hitlogic` *Hit Logic (Discrete)*

Description

Wrapper that computes discrete hitcalls for a provided PATHWAY_CR dataframe.

Usage

```
hitlogic(indf, newbmad = NULL, xs = NULL, ys = NULL, newcutoff = NULL)
```

Arguments

`indf`	Dataframe similar to PATHWAY_CR. Must contain "conc" and "resp" columns if xs and ys are not provided. Must contain "cutoff" and "bmad_factor" columns if newbmad is not NULL. Must contain "top" and "ac50" columns. "conc" and "resp" entries should be a single string with values separated by |.
`newbmad`	(Deprecated) New number of bmads to use for the cutoff.
`xs`	List of concentration vectors that can be provided for speed.
`ys`	List of response vectors that can be provided for speed.
`newcutoff`	Vector of new cutoff values to use. Length should be equal to rows in indf.

Value

Vector of hitcalls with length equal to number of rows in indf.

Examples

```
conc = rep(".03|.1|.3|1|3|10|30|100",2)
resp = rep("0|0|.1|.1|.5|.5|1|1",2)
indf = data.frame(top = c(1,1), ac50 = c(3,4), conc = conc, resp = resp,
  stringsAsFactors = FALSE)
hitlogic(indf, newcutoff = c(.8,1.2))
```

`hitloginner` *Hit Logic Inner (Discrete)*

Description

Contains hit logic, called directly during CR fitting or later through "hitlogic".

Usage

```
hitloginner(conc = NULL, resp, top, cutoff, ac50 = NULL)
```

Arguments

`conc`	Vector of concentrations (No longer necessary).
`resp`	Vector of responses.
`top`	Model top.
`cutoff`	Desired cutoff.
`ac50`	Model AC50 (No longer necessary).

Details

The purpose of this function is to keep the actual hit rules in one location so it can be called during CR fitting, and then again after the fact for a variety of cutoffs. Curves fit with constant winning should have top = NA, generating a miss.

Value

Outputs 1 for hit, 0 for miss.

Examples

```
hitloginner(resp = 1:8, top = 7, cutoff = 5) #hit
hitloginner(resp = 1:8, top = 7, cutoff = 7.5) #miss: top too low
hitloginner(resp = 1:8, top = 9, cutoff = 8.5) #miss: no response> cutoff
hitloginner(resp = 1:8, top = NA, cutoff = 5) #miss: no top (constant)
```

`loggnls` *Log Gain-Loss Model*

Description

Log Gain-Loss Model

Usage

```
loggnls(ps, x)
```

Arguments

ps	Vector of parameters: tp,ga,p,la,q,er
x	Vector of concentrations (log10 units)

Value

Vector of model responses

`loghill` *Log Hill Model*

Description

Log Hill Model

Usage

```
loghill(ps, x)
```

Arguments

ps	Vector of parameters: tp,ga,p,er
x	Vector of concentrations (log10 units)

Value

Vector of model responses

nestselect *Nest Select*

Description

Chooses between nested models.

Usage

```
nestselect(aics, mod1, mod2, dfdiff, pval = 0.05)
```

Arguments

aics	Named vector of model aics (can include extra models).
mod1	Name of model 1, the model with fewer degrees of freedom.
mod2	Name of model 2, the model with more degrees of freedom.
dfdiff	Absolute difference in number of degrees of freedom (i.e. the difference in parameters).
pval	P-value for nested model test.

Value

Named aic vector with losing model removed.

Examples

```
aics = c(-5,-6,-3)
names(aics) = c("poly1", "poly2", "hill")
nestselect(aics, "poly1", "poly2", 1)

aics = c(-5,-7,-3)
names(aics) = c("poly1", "poly2", "hill")
nestselect(aics, "poly1", "poly2", 1)
```

`poly1` *Polynomial 1 Model*

Description

Polynomial 1 Model

Usage

```
poly1(ps, x)
```

Arguments

`ps`	Vector of parameters: a,er
`x`	Vector of concentrations (regular units)

Value

Vector of model responses

`poly2` *Polynomial 2 Model*

Description

Polynomial 2 Model

Usage

```
poly2(ps, x)
```

Arguments

`ps`	Vector of parameters: a,b,er
`x`	Vector of concentrations (regular units)

Value

Vector of model responses

pow *Power Model*

Description

Power Model

Usage

```
pow(ps, x)
```

Arguments

ps	Vector of parameters: a,p,er
x	Vector of concentrations (regular units)

Value

Vector of model responses

tcplFit2.test *Run a test of the tcploFit2 code*

Description

This is jsut a test routine to show how tcplFit2 runs

Usage

```
tcplFit2.test()
```

Details

concRespCore(row, conthits = TRUE) concRespCorec(row, aicc = TRUE)

tcplObj *Concentration Response Objective Function*

Description

Log-likelihood to be maximized during CR fitting.

Usage

```
tcplObj(p, conc, resp, fname, errfun = "dt4", err = NULL)
```

Arguments

p	Vector of parameters, must be in order: a, tp, b, ga, p, la, q, er. Does not require names.
conc	Vector of concentrations in log10 units for loghill/loggnls, in regular units otherwise.
resp	Vector of corresponding responses.
fname	Name of model function.
errfun	Which error distribution to assume for each point. "dt4" is the original 4 degrees of freedom t-distribution. "dnorm" is the normal distribution.
err	An optional estimation of error for the given fit.

Details

This function is a generalized version of the log-likelihood estimation functions used in the ToxCast Pipeline (TCPL). Hill model uses fname "loghill" and gnls uses fname "loggnls". Other model functions have the same fname as their model name; i.e. exp2 uses "exp2", etc. errfun = "dnorm" may be better suited to gsva pathway scores than "dt4". Setting err could be used to fix error based on the null data noise distribution instead of fitting the error when maximizing log-likelihood.

Value

Log-likelihood.

Examples

```
conc = c(.03,.1 , .3 , 1  , 3 , 10 , 30  , 100)
resp = c( 0 , 0 , .1 ,.2 , .5 , 1  , 1.5 , 2  )
p = c(tp = 2, ga = 3, p = 4, er = .5)
tcplObj(p,conc,resp,"exp5")

lconc = log10(conc)
tcplObj(p,lconc,resp,"loghill")
```

toplikelihood *Top Likelihood*

Description

Probability of top being above cutoff.

Usage

```
toplikelihood(fname, cutoff, conc, resp, ps, top, mll)
```

Arguments

fname	Model function name (equal to model name except hill which uses "hillfn")
cutoff	Desired cutoff.
conc	Vector of concentrations.
resp	Vector of responses.
ps	Vector of parameters, must be in order: a, tp, b, ga, p, la, q, er
top	Model top.
mll	Winning model maximum log-likelihood.

Details

Should only be called by hitcontinner. Uses profile likelihood, similar to bmdbounds. Here, the y-scale type parameter is substituted in such a way that the top equals the cutoff. Then the log-likelihood is compared to the maximum log-likelihood using chisq function to retrieve probability.

Value

Probability of top being above cutoff.

Examples

```
fname = "hillfn"
conc = c(.03,.1,.3,1,3,10,30,100)
resp = c(0,.1,0,.2,.6,.9,1.1,1)
ps = c(1.033239, 2.453014, 1.592714, er = -3.295307)
top = 1.023239
mll = 12.71495
toplikelihood(fname, cutoff = .8, conc, resp, ps, top, mll)
toplikelihood(fname, cutoff = 1, conc, resp, ps, top, mll)
toplikelihood(fname, cutoff = 1.2, conc, resp, ps, top, mll)
```

Index