

Package ‘statmod’

February 15, 2012

Version 1.4.14

Date 2011/11/19

Title Statistical Modeling

Author Gordon Smyth with contributions from Yifang Hu, Peter Dunn and Belinda Phipson.

Maintainer Gordon Smyth <smyth@wehi.edu.au>

Depends R (>= 1.6.1)

Suggests MASS, tweedie

Description Various statistical modeling functions including growth curve comparisons, limiting dilution analysis, mixed linear models, heteroscedastic regression, Tweedie family generalized linear models, the inverse-Gaussian distribution and Gauss quadrature.

License LGPL (>= 2)

URL <http://www.statsci.org/r>

Repository CRAN

Date/Publication 2011-11-21 08:06:24

R topics documented:

1.StatMod	2
deprecated	3
Digamma	4
elda	6
fitNBP	8
gauss.quad	10
gauss.quad.prob	11
glm.scoretest	13
glnmgam.fit	14
growthcurve	15

hommel.test	17
invgauss	18
logmdigamma	19
matvec	20
meanT	21
mixedModel2	22
mscale	24
permp	25
plot.limdil	26
power.fisher.test	27
qresiduals	28
remlscore	30
remlscoregamma	31
sage.test	33
tweedie	34
welding	36
Index	38

Description

This library packages together those functions, other than those for microarray data analysis, which I wish to make public. A change-log for this package is available from <http://www.statsci.org/r/change-log.txt>.

Contributions to this library have also been made by Paul Bagshaw, Centre National d'Etudes des Telecommunications (DIH/DIPS), France (qinvgauss), and Trevor Park, Department of Statistics, University of Florida (rinvgauss).

Generalized Linear Models

tweedie, canonic.digamma, unitdeviance.digamma, varfun.digamma, cumulant.digamma, d2cumulant.digamma, meanval.digamma and logmdigamma are functions to fit non-standard generalized linear models related to the gamma distribution.

qres implements randomized quantile residuals for generalized linear models.

Growth Curves

compareGrowthCurves, compareTwoGrowthCurves and meanT are functions to test for differences between growth curves with repeated measurements on subjects.

Limiting Dilution Analysis

limdil implements limiting dilution analysis using complementary log-log binomial generalized linear model regression, with some improvements on previous programs.

Probability Distributions

`qinvgauss`, `dinvgauss`, `pinvgauss` and `rinvgauss` perform probability calculations for the inverse Gaussian distribution.

`gauss.quad` and `gauss.quad.prob` compute Gaussian Quadrature with probability distributions.

Tests

`hommel.test` performs Hommel's multiple comparison tests.

`power.fisher.test` computes the power of Fisher's Exact Test for comparing proportions.

`sage.test` is a fast approximation to Fisher's exact test for each tag for comparing two Serial Analysis of Gene Expression (SAGE) libraries.

Variance Models

`mixedModel2`, `mixedModel2Fit` and `glmGam.fit` fit mixed linear models.

`remlscore` and `remlscoregamma` fit heteroscedastic and varying dispersion models by REML. `welding` is an example data set.

Matrix Computations

`matvec` and `vecmat` facilitate multiplying matrices by vectors.

Author(s)

Gordon Smyth

deprecated

Deprecated Functions in statmod Package

Description

These functions are provided for compatibility with older versions of R only, and may be defunct as soon as the next release.

Usage

```
randomizedBlock(formula, random, weights=NULL, only.varcomp=FALSE, data=list(), subset=NULL, contrast)
randomizedBlockFit(y, X, Z, w=NULL, only.varcomp=FALSE, tol=1e-6, maxit=50, trace=FALSE)
```

Arguments

	The arguments <code>formula</code> , <code>weights</code> , <code>data</code> , <code>subset</code> and <code>contrasts</code> have the same meaning as in <code>lm</code> . The arguments <code>y</code> , <code>X</code> and <code>w</code> have the same meaning as in <code>lm.wfit</code> .
	formula specifying the fixed model.
<code>random</code>	vector or factor specifying the blocks corresponding to random effects.
<code>weights</code>	optional vector of prior weights.
<code>only.varcomp</code>	logical value, if TRUE computation of standard errors and fixed effect coefficients will be skipped
<code>data</code>	an optional data frame containing the variables in the model.
<code>subset</code>	an optional vector specifying a subset of observations to be used in the fitting process.
<code>contrasts</code>	an optional list. See the <code>contrasts.arg</code> argument of <code>model.matrix.default</code> .
<code>tol</code>	small positive numeric tolerance, passed to <code>glmGam.fit</code>
<code>maxit</code>	maximum number of iterations permitted, passed to <code>glmGam.fit</code>
<code>trace</code>	logical value, passed to <code>glmGam.fit</code> . If TRUE then working estimates will be printed at each iteration.
<code>y</code>	numeric response vector
<code>X</code>	numeric design matrix for fixed model
<code>Z</code>	numeric design matrix for random effects
<code>w</code>	optional vector of prior weights

Details

`randomizedBlock` and `randomizedBlockFit` have been renamed to `mixedModel2` and `mixedModel2Fit` respectively.

 Digamma

Digamma generalized linear model family

Description

Produces a Digamma generalized linear model family object. The Digamma distribution is the distribution of the unit deviance for a gamma response.

Usage

```

Digamma(link = "log")
unitdeviance.digamma(y, mu)
cumulant.digamma(theta)
meanval.digamma(theta)
d2cumulant.digamma(theta)
varfun.digamma(mu)
canonic.digamma(mu)

```

Arguments

link	character string, number or expression specifying the link function. See <code>quasi</code> for specification of this argument.
y	numeric vector of (positive) response values
mu	numeric vector of (positive) fitted values
theta	numeric vector of values of the canonical variable, equal to $-1/\phi$ where ϕ is the dispersion parameter of the gamma distribution

Details

This family is useful for dispersion modelling with gamma generalized linear models. The Digamma distribution describes the distribution of the unit deviances for a gamma family, in the same way that the gamma distribution itself describes the distribution of the unit deviances for Gaussian or inverse Gaussian families. The Digamma distribution is so named because it is dual to the gamma distribution in the above sense, and because the `digamma` function appears in its mean function.

Suppose that y follows a gamma distribution with mean μ and dispersion parameter ϕ , so the variance of y is $\phi\mu^2$. Write $d(y, \mu)$ for the gamma distribution unit deviance. Then `meanval.digamma(-1/phi)` gives the mean of $d(y, \mu)$ and `2*d2cumulant.digamma(-1/phi)` gives the variance.

Value

`Digamma` produces a `glm` family object, which is a list of functions and expressions used by `glm` in its iteratively reweighted least-squares algorithm. See `family` for details.

The other functions take vector arguments and produce vector values of the same length and called by `Digamma`. `unitdeviance.digamma` gives the unit deviances of the family, equal to the squared deviance residuals. `cumulant.digamma` is the cumulant function. If the dispersion is unity, then successive derivatives of the cumulant function give successive cumulants of the Digamma distribution. `meanvalue.digamma` gives the first derivative, which is the expected value. `d2cumulant.digamma` gives the second derivative, which is the variance. `canonic.digamma` is the inverse of `meanvalue.digamma` and gives the canonical parameter as a function of the mean parameter. `varfun.digamma` is the variance function of the Digamma family, the variance as a function of the mean.

Author(s)

Gordon Smyth

References

Smyth, G. K. (1989). Generalized linear models with varying dispersion. *J. R. Statist. Soc. B*, **51**, 47-61.

See Also

[quasi](#), [make.link](#)

Examples

```
# Test for log-linear dispersion trend in gamma regression
y <- rchisq(20,df=1)
x <- 1:20
out.gam <- glm(y~x,family=Gamma(link="log"))
d <- residuals(out.gam)^2
out.dig <- glm(d~x,family=Digamma(link="log"))
summary(out.dig,dispersion=2)
```

elda

Extreme Limiting Dilution Analysis

Description

Fit single-hit model to a dilution series using complementary log-log binomial regression.

Usage

```
elda(response, dose, tested=rep(1,length(response)), group=rep(1,length(response)), observed=FALSE, confidence,
limdil(response, dose, tested=rep(1,length(response)), group=rep(1,length(response)), observed=FALSE, confidence)
```

Arguments

response	numeric of integer counts of positive cases, out of tested trials
dose	numeric vector of expected number of cells in assay
tested	numeric vector giving number of trials at each dose
group	vector or factor giving group to which the response belongs
observed	logical, is the actual number of cells observed?
confidence	numeric level for confidence interval
test.unit.slope	logical, should the adequacy of the single-hit model be tested?

Details

elda and limdil are alternative names for the same function. limdil was the older name before the publication Hu and Smyth (2009).

This function is an implementation of maximum likelihood analysis of limiting dilution data with added features to accommodate small sample sizes (Hu and Smyth, 2009). In particular, the function accommodates gracefully situations where 0% or 100% of the assays give positive results, which is why we call it "extreme" limiting dilution analysis. The methodology has been applied to the analysis of stem cell assays (Shackleton et al, 2006).

A binomial generalized linear model is fitted for each group with cloglog link and offset $\log(\text{dose})$. If observed=FALSE, a classic Poisson single-hit model is assumed, and the Poisson frequency of the stem cells is the exp of the intercept. If observed=TRUE, the values of dose are treated as actual cell numbers rather than expected values. This doesn't changed the generalized linear model fit but

changes how the frequencies are extracted from the estimated model coefficient (Hu and Smyth, 2009).

The confidence interval is a Wald confidence interval, unless all the responses are zero or at the maximum value, in which case Clopper-Pearson intervals are computed.

If group takes several values, then separate confidence intervals are computed for each group. In this case it also possible to test for non-equality in frequency between the groups.

These functions produce objects of class "limdil". There are `print` and `plot` methods for "limdil" objects.

Value

An object of class "limdil" with the following components:

CI	numeric vector giving estimated frequency and lower and upper limits of Wald confidence interval of each group
test.difference	numeric vector giving chisquare likelihood ratio test statistic and p-value for testing the difference between groups
test.slope.wald	numeric vector giving wald test statistics and p-value for testing the slope of the offset equal to one
test.slope.lr	numeric vector giving chisquare likelihood ratio test statistics and p-value for testing the slope of the offset equal to one
test.slope.score1	numeric vector giving score test statistics and p-value for testing multi-hit alternatives
test.slope.score	numeric vector giving score test statistics and p-value for testing heterogeneity
response	numeric of integer counts of positive cases, out of tested trials
tested	numeric vector giving number of trials at each dose
dose	numeric vector of expected number of cells in assay
group	vector or factor giving group to which the response belongs
num.group	number of groups

Author(s)

Yifang Hu and Gordon Smyth

References

Shackleton, M., Vaillant, F., Simpson, K. J., Stingl, J., Smyth, G. K., Asselin-Labat, M.-L., Wu, L., Lindeman, G. J., and Visvader, J. E. (2006). Generation of a functional mammary gland from a single stem cell. *Nature* 439, 84-88. <http://www.nature.com/nature/journal/v439/n7072/abs/nature04372.html>

Hu, Y, and Smyth, GK (2009). ELDA: Extreme limiting dilution analysis for comparing depleted and enriched populations in stem cell and other assays. *Journal of Immunological Methods* 347, 70-78. <http://dx.doi.org/10.1016/j.jim.2009.06.008>

See Also

A webpage interface to this function is available at <http://bioinf.wehi.edu.au/software/elda>.

Examples

```
# When there is one group
Dose <- c(50,100,200,400,800)
Responses <- c(2,6,9,15,21)
Tested <- c(24,24,24,24,24)
out <- elda(Responses,Dose,Tested,test.unit.slope=TRUE)
out
plot(out)

# When there are four groups
Dose <- c(30000,20000,4000,500,30000,20000,4000,500,30000,20000,4000,500,30000,20000,4000,500)
Responses <- c(2,3,2,1,6,5,6,1,2,3,4,2,6,6,6,1)
Tested <- c(6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6)
Group <- c(1,1,1,1,2,2,2,2,3,3,3,3,4,4,4,4)
elda(Responses,Dose,Tested,Group,test.unit.slope=TRUE)
```

fitNBP

Negative Binomial Model for SAGE Libraries with Pearson Estimation of Dispersion

Description

Fit a multi-group negative-binomial model to SAGE data, with Pearson estimation of the common overdispersion parameter.

Usage

```
fitNBP(y, group=NULL, lib.size=colSums(y), tol=1e-5, maxit=40, verbose=FALSE)
```

Arguments

y	numeric matrix giving counts. Rows correspond to tags (genes) and columns to SAGE libraries.
group	factor indicating which library belongs to each group. If NULL then one group is assumed.
lib.size	vector giving total number of tags in each library.
tol	small positive numeric tolerance to judge convergence
maxit	maximum number of iterations permitted
verbose	logical, if TRUE then iteration progress information is output.

Details

The overdispersion parameter is estimated equating the Pearson goodness of fit to its expectation. The variance is assumed to be of the form $\text{Var}(y)=\mu*(1+\phi*\mu)$ where $E(y)=\mu$ and ϕ is the dispersion parameter. All tags are assumed to share the same dispersion.

For given dispersion, the model for each tag is a negative-binomial generalized linear model with log-link and `log(lib.size)` as offset. The coefficient parametrization used is that corresponding to the formula `~0+group+offset(log(lib.size))`.

Except for the dispersion being common rather than genewise, the model fitted by this function is equivalent to that proposed by Lu et al (2005). The numeric algorithm used is that of alternating iterations (Smyth, 1996) using Newton's method as the outer iteration for the dispersion parameter starting at $\phi=0$. This iteration is monotonically convergent for the dispersion.

Value

List with components

<code>coefficients</code>	numeric matrix of rates for each tag (gene) and each group
<code>fitted.values</code>	numeric matrix of fitted values
<code>dispersion</code>	estimated dispersion parameter

Author(s)

Gordon Smyth

References

Lu, J, Tomfohr, JK, Kepler, TB (2005). Identifying differential expression in multiple SAGE libraries: an overdispersed log-linear model approach. *BMC Bioinformatics* 6,165.

Smyth, G. K. (1996). Partitioned algorithms for maximum likelihood and other nonlinear estimation. *Statistics and Computing*, 6, 201-216.

See Also

[sage.test](#)

Examples

```
# True value for dispersion is 1/size=2/3
# Note the Pearson method tends to under-estimate the dispersion
y <- matrix(rnbinom(10*4,mu=4,size=1.5),10,4)
lib.size <- rep(50000,4)
group <- c(1,1,2,2)
fit <- fitNBP(y,group=group,lib.size=lib.size)
logratio <- fit$coef %*% c(-1,1)
```

gauss.quad

*Gaussian Quadrature***Description**

Calculate nodes and weights for Gaussian quadrature.

Usage

```
gauss.quad(n,kind="legendre",alpha=0,beta=0)
```

Arguments

n	number of nodes and weights
kind	kind of Gaussian quadrature, one of "legendre", "chebyshev1", "chebyshev2", "hermite", "jacobi" or "laguerre"
alpha	parameter for Jacobi or Laguerre quadrature, must be greater than -1
beta	parameter for Jacobi quadrature, must be greater than -1

Details

The integral from a to b of $w(x)*f(x)$ is approximated by $\text{sum}(w*f(x))$ where x is the vector of nodes and w is the vector of weights. The approximation is exact if f(x) is a polynomial of order no more than $2n-1$. The possible choices for w(x), a and b are as follows:

Legendre quadrature: $w(x)=1$ on $(-1, 1)$.

Chebyshev quadrature of the 1st kind: $w(x)=1/\text{sqrt}(1-x^2)$ on $(-1, 1)$.

Chebyshev quadrature of the 2nd kind: $w(x)=\text{sqrt}(1-x^2)$ on $(-1, 1)$.

Hermite quadrature: $w(x)=\text{exp}(-x^2)$ on $(-\text{Inf}, \text{Inf})$.

Jacobi quadrature: $w(x)=(1-x)^\alpha*(1+x)^\beta$ on $(-1, 1)$. Note that Chebyshev quadrature is a special case of this.

Laguerre quadrature: $w(x)=x^\alpha*\text{exp}(-x)$ on $(0, \text{Inf})$.

The method is explained in Golub and Welsch (1969).

Value

A list containing the components

nodes	vector of values at which to evaluate the function
weights	vector of weights to give the function values

Note

This function solves a dense $n \times n$ eigenvector problem and is therefore slow for large n. It could be made far more efficient by using an eigenvector function designed to compute the leading terms of the eigenvectors for tridiagonal matrices.

Author(s)

Gordon Smyth

References

Golub, G. H., and Welsch, J. H. (1969). Calculation of Gaussian quadrature rules. *Mathematics of Computation* **23**, 221-230.

Golub, G. H. (1973). Some modified matrix eigenvalue problems. *Siam Review* **15**, 318-334.

Stroud and Secrest (1966). *Gaussian Quadrature Formulas*. Prentice- Hall, Englewood Cliffs, N.J.

See Also

[gauss.quad.prob](#), [integrate](#)

Examples

```
out <- gauss.quad(10,"laguerre",alpha=5)
sum(out$weights * out$nodes) / gamma(6)
# mean of gamma distribution with alpha=6
```

 gauss.quad.prob

Gaussian Quadrature with Probability Distributions

Description

Calculate nodes and weights for Gaussian quadrature in terms of probability distributions.

Usage

```
gauss.quad.prob(n,dist="uniform",l=0,u=1,mu=0,sigma=1,alpha=1,beta=1)
```

Arguments

n	number of nodes and weights
dist	distribution that Gaussian quadrature is based on, one of "uniform", "normal", "beta" or "gamma"
l	lower limit of uniform distribution
u	upper limit of uniform distribution
mu	mean of normal distribution
sigma	standard deviation of normal distribution
alpha	positive shape parameter for gamma distribution or first shape parameter for beta distribution
beta	positive scale parameter for gamma distribution or second shape parameter for beta distribution

Details

This is a rewriting and simplification of `gauss.quad` in terms of probability distributions.

The expected value of $f(X)$ is approximated by $\sum(w*f(x))$ where x is the vector of nodes and w is the vector of weights. The approximation is exact if $f(x)$ is a polynomial of order no more than $2n-1$. The possible choices for the distribution of X are as follows:

Uniform on $(1, u)$.

Normal with mean μ and standard deviation σ .

Beta with density $x^{(\alpha-1)}*(1-x)^{(\beta-1)}/B(\alpha, \beta)$ on $(0, 1)$.

Gamma with density $x^{(\alpha-1)}*\exp(-x/\beta)/\beta^\alpha/\text{gamma}(\alpha)$.

Value

A list containing the components

nodes vector of values at which to evaluate the function

weights vector of weights to give the function values

Author(s)

Gordon Smyth

References

Golub, G. H., and Welsch, J. H. (1969). Calculation of Gaussian quadrature rules. *Mathematics of Computation* **23**, 221-230.

Golub, G. H. (1973). Some modified matrix eigenvalue problems. *Siam Review* **15**, 318-334.

Smyth, G. K. (1998). Polynomial approximation. In: *Encyclopedia of Biostatistics*, P. Armitage and T. Colton (eds.), Wiley, London, pp. 3425-3429. <http://www.statsci.org/smyth/pubs/poly.ps>

Stroud and Secrest (1966). *Gaussian Quadrature Formulas*. Prentice- Hall, Englewood Cliffs, N.J.

See Also

[gauss.quad](#), [integrate](#)

Examples

```
out <- gauss.quad.prob(10, "normal")
sum(out$weights * out$nodes^4)
# the 4th moment of the standard normal is 3

out <- gauss.quad.prob(32, "gamma", alpha=5)
sum(out$weights * log(out$nodes))
# the expected value of log(X) where X is gamma is digamma(alpha)
```

`glm.scoretest`*Score Test for Adding a Covariate to a GLM*

Description

Computes score test statistics (z-statistics) for adding covariates to a generalized linear model.

Usage

```
glm.scoretest(fit, x2, dispersion=NULL)
```

Arguments

<code>fit</code>	generalized linear model fit object, of class <code>glm</code> .
<code>x2</code>	vector or matrix with each column a covariate to be added.
<code>dispersion</code>	the dispersion for the generalized linear model family.

Details

Rao's score statistic. Is the locally most powerful test for testing vs a one-sided alternative. Asymptotically equivalent to likelihood ratio tests, but convenient for one-sided tests.

This function computes a score test statistics for adding each covariate individually.

The dispersion parameter is treated as for `summary.glm`. If NULL, the Pearson estimator is used, except for the binomial and Poisson families, for which the dispersion is one.

Value

numeric vector containing the z-statistics, one for each covariate.

Author(s)

Gordon Smyth

References

Lovison, G (2005). On Rao score and Pearson χ^2 statistics in generalized linear models. *Statistical Papers*, 46, 555-574.

Pregibon, D (1982). Score tests in GLIM with applications. In *GLIM82: Proceedings of the International Conference on Generalized Linear Models*, R Gilchrist (ed.), Lecture Notes in Statistics, Volume 14, Springer, New York, pages 87-97.

Smyth, G. K. (2003). Pearson's goodness of fit statistic as a score test statistic. In: *Science and Statistics: A Festschrift for Terry Speed*, D. R. Goldstein (ed.), IMS Lecture Notes - Monograph Series, Volume 40, Institute of Mathematical Statistics, Beachwood, Ohio, pages 115-126. <http://www.statsci.org/smyth/pubs/goodness.pdf>

See Also[glm](#), [add1](#)**Examples**

```
# Pearson's chisquare test for independence
# in a contingency table is a score test.

# First the usual test

y <- c(20,40,40,30)
chisq.test(matrix(y,2,2),correct=FALSE)

# Now same test using glm.scoretest

a <- gl(2,1,4)
b <- gl(2,2,4)
fit <- glm(y~a+b,family=poisson)
x2 <- c(0,0,0,1)
z <- glm.scoretest(fit,x2)
z^2
```

glmGam.fit

Fit Generalized Linear Model by Fisher Scoring with Levenberg Damping

Description

Fit a generalized linear model with secure convergence. Provided for gamma glm with identity links or negative binomial glm with log-links.

Usage

```
glmGam.fit(X, y, start=NULL, tol=1e-6, maxit=50, trace=FALSE)
glmnb.fit(X, y, dispersion, offset=0, start=NULL, tol=1e-6, maxit=50, trace=FALSE)
```

Arguments

X	design matrix, assumed to be of full column rank. Missing values not allowed.
y	numeric vector of responses. Negative or missing values not allowed.
dispersion	numeric vector of over-dispersion parameters for negative binomial. If of length 1, then same over-dispersion is assumed for all observations.
offset	offset vector for linear model
start	numeric vector of starting values for the regression coefficients
tol	small positive numeric value giving convergence tolerance
maxit	maximum number of iterations allowed
trace	logical value. If TRUE then output diagnostic information at each iteration.

Details

These functions implement a modified Fisher scoring algorithm for generalized linear models, similar to the Levenberg-Marquardt algorithm for nonlinear least squares. The Levenberg-Marquardt modification checks for a reduction in the deviance at each step, and avoids the possibility of divergence. The result is a very secure algorithm that converges for almost all datasets.

`glmGam.fit` is in principle similar to `glm.fit(X,y,family=Gamma(link="identity"))` but with much more secure convergence. This function is used by [mixedModel2Fit](#).

`glmnb.fit` is in principle similar to `glm.fit(X,y,family=negative.binomial(link="log",theta=1/dispersion))` but with more secure convergence.

Value

List with the following components:

<code>coefficients</code>	numeric vector of regression coefficients
<code>fitted</code>	numeric vector of fitted values
<code>deviance</code>	residual deviance
<code>iter</code>	number of iterations used to convergence. If convergence was not achieved then <code>iter</code> is set to <code>maxit+1</code> .

Author(s)

Gordon Smyth and Yunshun Chen

Examples

```
y <- rgamma(10,shape=5)
X <- cbind(1,1:10)
fit <- glmGam.fit(X,y,trace=TRUE)
```

growthcurve

Compare Groups of Growth Curves

Description

Do all pairwise comparisons between groups of growth curves using a permutation test.

Usage

```
compareGrowthCurves(group,y,levels=NULL,nsim=100,fun=meanT,times=NULL,verbose=TRUE,adjust="holm")
compareTwoGrowthCurves(group,y,nsim=100,fun=meanT)
plotGrowthCurves(group,y,levels=sort(unique(group)),times=NULL,col=NULL,...)
```

Arguments

group	vector or factor indicating group membership. Missing values are allowed in <code>compareGrowthCurves</code> but not in <code>compareTwoGrowthCurves</code> .
y	matrix of response values with rows for individuals and columns for times. The number of rows must agree with the length of <code>group</code> . Missing values are allowed.
levels	a character vector containing the identifiers of the groups to be compared. By default all groups with two or more members will be compared.
nsim	number of permutations to estimate p-values.
fun	the statistic used to measure the distance between two groups of growth curves. Default to a mean t-statistic.
times	a numeric vector containing the column numbers on which the groups should be compared. By default all the columns are used.
verbose	should progress results be printed?
adjust	method used to adjust for multiple testing, see <code>p.adjust</code> .
col	vector of colors corresponding to distinct groups
...	other arguments passed to <code>plot()</code>

Details

`compareTwoGrowthCurves` performs a permutation test of the difference between two groups of growth curves. `compareGrowthCurves` does all pairwise comparisons between two or more groups of growth curves. Accurate p-values can be obtained by setting `nsim` to some large value, `nsim=10000` say.

Value

`compareTwoGrowthCurves` returns a list with two components, `stat` and `p.value`, containing the observed statistics and the estimated p-value. `compareGrowthCurves` returns a data frame with components

Group1	name of first group in a comparison
Group2	name of second group in a comparison
Stat	observed value of the statistic
P.Value	estimated p-value
adj.P.Value	p-value adjusted for multiple testing

Author(s)

Gordon Smyth

References

Elso, C. M., Roberts, L. J., Smyth, G. K., Thomson, R. J., Baldwin, T. M., Foote, S. J., and Handman, E. (2004). Leishmaniasis host response loci (lmr13) modify disease severity through a Th1/Th2-independent pathway. *Genes and Immunity* 5, 93-100. <http://www.nature.com/gene/journal/v5/n2/full/6364042a.html>

Baldwin, T., Sakthianandeswaren, A., Curtis, J., Kumar, B., Smyth, G. K., Foote, S., and Handman, E. (2007). Wound healing response is a major contributor to the severity of cutaneous leishmaniasis in the ear model of infection. *Parasite Immunology* 29, 501-513. <http://www.blackwell-synergy.com/doi/abs/10.1111/j.1365-3024.2007.00969.x>

See Also

[compareGrowthCurves](#), [compareTwoGrowthCurves](#)

Examples

```
# A example with only one time
data(PlantGrowth)
compareGrowthCurves(PlantGrowth$group,as.matrix(PlantGrowth$weight))
# Can make p-values more accurate by nsim=10000
```

hommel.test

Test Multiple Comparisons Using Hommel's Method

Description

Given a set of p-values and a test level, returns vector of test results for each hypothesis.

Usage

```
hommel.test(p, alpha=0.05)
```

Arguments

p	numeric vector of p-values
alpha	numeric value, desired significance level

Details

This function implements the multiple testing procedure of Hommel (1988). Hommel's method is also implemented as an adjusted p-value method in the function `p.adjust` but the accept/reject approach used here is faster.

Value

logical vector indicating whether each hypothesis is accepted

Author(s)

Gordon Smyth

References

Hommel, G. (1988). A stagewise rejective multiple test procedure based on a modified Bonferroni test. *Biometrika*, **75**, 383-386.

Shaffer, J. P. (1995). Multiple hypothesis testing. *Annual Review of Psychology* **46**, 561-576. (An excellent review of the area.)

See Also

[p.adjust](#)

Examples

```
p <- sort(runif(100))[1:10]
cbind(p,p.adjust(p,"hommel"),hommel.test(p))
```

 invgauss

Inverse Gaussian Distribution

Description

Density, cumulative probability, quantiles and random generation for the inverse Gaussian distribution.

Usage

```
dinvgauss(x, mu, lambda=1, log=FALSE)
pinvgauss(q, mu, lambda=1)
qinvgauss(p, mu, lambda=1)
rinvgauss(n, mu, lambda=1)
```

Arguments

x	vector of quantiles. Missing values (NAs) are allowed.
q	vector of quantiles. Missing values (NAs) are allowed.
p	vector of probabilities. Missing values (NAs) are allowed.
n	sample size. If length(n) is larger than 1, then length(n) random values are returned.
mu	vector of (positive) means. This is replicated to be the same length as p or q or the number of deviates generated.
lambda	vector of (positive) precision parameters. This is replicated to be the same length as p or q or the number of deviates generated.
log	logical; if TRUE, the log-density is returned.

Details

The inverse Gaussian distribution takes values on the positive real line. The variance of the distribution is μ^3/λ . Applications of the inverse Gaussian include sequential analysis, diffusion processes and radiotechniques. The inverse Gaussian is one of the response distributions used in generalized linear models.

Value

Vector of same length as `x` or `q` giving the density (`dinvgauss`), probability (`pinvgauss`), quantile (`qinvgauss`) or random sample (`rinvgauss`) for the inverse Gaussian distribution with mean `mu` and inverse dispersion `lambda`. Elements of `q` or `p` that are missing will cause the corresponding elements of the result to be missing.

Author(s)

Gordon Smyth; Paul Bagshaw, Centre National d'Etudes des Telecommunications (DIH/DIPS), France (`qinvgauss`); Trevor Park, Department of Statistics, University of Florida

References

Chhikara, R. S., and Folks, J. Leroy, (1989). *The inverse Gaussian distribution: Theory, methodology, and applications*. Marcel Dekker, New York.

See Also

`dinvGauss`, `pinvGauss`, `qinvGauss` and `rinvGauss` in the `SuppDists` package.

Examples

```
y <- rinvgauss(20,1,2) # generate vector of 20 random numbers
p <- pinvgauss(y,1,2) # p should be uniform
```

logmdigamma

Log Minus Digamma Function

Description

The difference between the log and digamma functions.

Usage

```
logmdigamma(x)
```

Arguments

`x` numeric vector or array of positive values. Negative or zero values will return NA.

Details

`digamma(x)` is asymptotically equivalent to $\log(x)$. `logmdigamma(x)` computes $\log(x) - \text{digamma}(x)$ without subtractive cancellation for large x .

Author(s)

Gordon Smyth

References

Abramowitz, M., and Stegun, I. A. (1970). *Handbook of mathematical functions*. Dover, New York.

See Also

[digamma](#)

Examples

```
log(10^15) - digamma(10^15) # returns 0
logmdigamma(10^15) # returns value correct to 15 figures
```

matvec

Multiply a Matrix by a Vector

Description

Multiply the rows or columns of a matrix by the elements of a vector.

Usage

```
matvec(M, v)
vecmat(v, M)
```

Arguments

<code>M</code>	numeric matrix, or object which can be coerced to a matrix.
<code>v</code>	numeric vector, or object which can be coerced to a vector. Length should match the number of columns of <code>M</code> (for <code>matvec</code>) or the number of rows of <code>M</code> (for <code>vecmat</code>)

Details

`matvec(M,v)` is equivalent to `M %*% diag(v)` but is faster to execute. Similarly `vecmat(v,M)` is equivalent to `diag(v) %*% M` but is faster to execute.

Value

A matrix of the same dimensions as `M`.

Author(s)

Gordon Smyth

Examples

```
A <- matrix(1:12,3,4)
A
matvec(A,c(1,2,3,4))
vecmat(c(1,2,3),A)
```

meanT*Mean t-Statistic Between Two Groups of Growth Curves*

Description

The mean-t statistic of the distance between two groups of growth curves.

Usage

```
meanT(y1, y2)
```

Arguments

y1 matrix of response values for the first group, with a row for each individual and a column for each time. Missing values are allowed.

y2 matrix of response values for the second group. Must have the same number of columns as y1. Missing values are allowed.

Details

This function computes the pooled two-sample t-statistic between the response values at each time, and returns the mean of these values weighted by the degrees of freedom. This function is used by `compareGrowthCurves`.

Value

numeric vector of length one containing the mean t-statistic.

Author(s)

Gordon Smyth

See Also

[compareGrowthCurves](#), [compareTwoGrowthCurves](#)

Examples

```

y1 <- matrix(rnorm(4*3),4,3)
y2 <- matrix(rnorm(4*3),4,3)
meanT(y1,y2)

data(PlantGrowth)
compareGrowthCurves(PlantGrowth$group,as.matrix(PlantGrowth$weight))
# Can make p-values more accurate by nsim=10000

```

mixedModel2

Fit Mixed Linear Model with 2 Error Components

Description

Fits a mixed linear model by REML. The linear model contains one random factor apart from the unit errors.

Usage

```

mixedModel2(formula, random, weights=NULL, only.varcomp=FALSE, data=list(), subset=NULL, contrasts=NULL)
mixedModel2Fit(y, X, Z, w=NULL, only.varcomp=FALSE, tol=1e-6, maxit=50, trace=FALSE)

```

Arguments

The arguments `formula`, `weights`, `data`, `subset` and `contrasts` have the same meaning as in `lm`. The arguments `y`, `X` and `w` have the same meaning as in `lm.wfit`.

<code>formula</code>	formula specifying the fixed model.
<code>random</code>	vector or factor specifying the blocks corresponding to random effects.
<code>weights</code>	optional vector of prior weights.
<code>only.varcomp</code>	logical value, if TRUE computation of standard errors and fixed effect coefficients will be skipped
<code>data</code>	an optional data frame containing the variables in the model.
<code>subset</code>	an optional vector specifying a subset of observations to be used in the fitting process.
<code>contrasts</code>	an optional list. See the <code>contrasts.arg</code> argument of <code>model.matrix.default</code> .
<code>tol</code>	small positive numeric tolerance, passed to <code>glmGam.fit</code>
<code>maxit</code>	maximum number of iterations permitted, passed to <code>glmGam.fit</code>
<code>trace</code>	logical value, passed to <code>glmGam.fit</code> . If TRUE then working estimates will be printed at each iteration.
<code>y</code>	numeric response vector
<code>X</code>	numeric design matrix for fixed model
<code>Z</code>	numeric design matrix for random effects
<code>w</code>	optional vector of prior weights

Details

This function fits the model $y = Xb + Zu + e$ where b is a vector of fixed coefficients and u is a vector of random effects. Write n for the length of y and q for the length of u . The random effect vector u is assumed to be normal, mean zero, with covariance matrix $\sigma_u^2 I_q$ while e is normal, mean zero, with covariance matrix $\sigma^2 I_n$. If Z is an indicator matrix, then this model corresponds to a randomized block experiment. The model is fitted using an eigenvalue decomposition which transforms the problem into a Gamma generalized linear model.

Note that the block variance component `varcomp[2]` is not constrained to be non-negative. It may take negative values corresponding to negative intra-block correlations. However the correlation `varcomp[2]/sum(varcomp)` must lie between -1 and 1.

Missing values in the data are not allowed.

This function is equivalent to `lme(fixed=formula,random=~1|random)`, except that the block variance component is not constrained to be non-negative, but is faster and more accurate for small to moderate size data sets. It is slower than `lme` when the number of observations is large.

This function tends to be fast and reliable, compared to competitor functions which fit randomized block models, when then number of observations is small, say no more than 200. However it becomes quadratically slow as the number of observations increases because of the need to do two eigenvalue decompositions of order nearly equal to the number of observations. So it is a good choice when fitting large numbers of small data sets, but not a good choice for fitting large data sets.

Value

A list with the components:

<code>varcomp</code>	vector of length two containing the residual and block components of variance.
<code>se.varcomp</code>	standard errors for the components of variance.
<code>reml.residuals</code>	standardized residuals in the null space of the design matrix.

If `fixed.estimated=TRUE` then the components from the diagonalized weighted least squares fit are also returned.

Author(s)

Gordon Smyth

References

Venables, W., and Ripley, B. (2002). *Modern Applied Statistics with S-Plus*, Springer.

See Also

[glmGam.fit](#), [lme](#), [lm](#), [lm.fit](#)

Examples

```
# Compare with first data example from Venable and Ripley (2002),
# Chapter 10, "Linear Models"
library(MASS)
data(petrol)
out <- mixedModel2(Y~SG+VP+V10+EP, random=No, data=petrol)
cbind(varcomp=out$varcomp,se=out$se.varcomp)
```

mscale

*M Scale Estimation***Description**

Robust estimation of a scale parameter using Hampel's redescending psi function.

Usage

```
mscale(u, na.rm=FALSE)
```

Arguments

`u` numeric vector of residuals.
`na.rm` logical. Should missing values be removed?

Details

Estimates a scale parameter or standard deviation using an M-estimator with 50% breakdown. This means the estimator is highly robust to outliers. If the input residuals `u` are a normal sample, then `mscale(u)` should be equal to the standard deviation.

Value

numeric constant giving the estimated scale.

Author(s)

Gordon Smyth

References

Yohai, V. J. (1987). High breakdown point and high efficiency robust estimates for regression. *Ann. Statist.* 15, 642-656.

Stromberg, A. J. (1993). Computation of high breakdown nonlinear regression parameters. *J. Amer. Statist. Assoc.* 88, 237-244.

Smyth, G. K., and Hawkins, D. M. (2000). Robust frequency estimation using elemental sets. *Journal of Computational and Graphical Statistics* 9, 196-214.

Examples

```
u <- rnorm(100)
sd(u)
mscale(u)
```

permp	<i>Exact permutation p-values</i>
-------	-----------------------------------

Description

Calculates exact p-values for permutation tests with permutations sampled with replacement.

Usage

```
permp(x, nperm, n1, n2, total.nperm=NULL, method="auto", twosided=TRUE)
```

Arguments

x	numeric vector or array giving the number of permutations that yield test statistics at least as extreme as that observed.
nperm	number of permutations performed.
n1	sample size of group 1. Not required if total.nperm is supplied.
n2	sample size of group 2. Not required if total.nperm is supplied.
total.nperm	total number of permutations allowable from the design of the experiment.
method	computation method, possible values are "exact", "approximate" or "auto".
twosided	logical, is the test two-sided so that the test statistic is symmetric between the two groups?

Details

This function can be used for calculating exact p-values for permutation tests where permutations are sampled with replacement, using theory and methods developed by Phipson and Smyth (2010). total.nperm is the total number of distinct values of the test statistic that are possible. This is generally equal to the number of possible permutations, unless a two-sided test is conducted with equal sample sizes, in which case total.nperm is half the number of permutations, because the test statistic must then be symmetric in the two groups. total.nperm can be supplied directly by the user, alternatively it will be computed from n1 and n2.

When method="exact", the p-values are computed to full machine precision by evaluating a summation. When method="approximate", an approximation is used that is faster and uses less memory. If method="auto", the exact calculation is used when total.nperm is less than or equal to 10,000, otherwise the approximation is used.

Value

vector or array of p-values, of same dimensions as x

Author(s)

Belinda Phipson and Gordon Smyth

References

Phipson, B and Smyth, GK (2010). Exact permutation p-values when permutations are randomly drawn. Technical Report, Walter and Eliza Hall Institute of Medical Research, Melbourne, Australia.

Examples

```
# Consider a two-sided permutation test with 99 permutations,
# 5 reaching the statistically significant threshold.
# Assume a two group experiment with 6 in each group.
# Input total.nperm=462

permp(x=5, nperm=99, total.nperm=462)

# Input n1=6 and n2=6

permp(x=5, nperm=99, n1=6, n2=6)

# Suppose we have a vector of p-values taking on the values 0 to 10 for the same experiment described above.

x<-0:10
permp(x=x, nperm=99, total.nperm=462)
```

plot.limdil

Plot or print an object of class limdil

Description

Plot or print an object of class limdil.

Usage

```
## S3 method for class 'limdil'
print(x, ...)
## S3 method for class 'limdil'
plot(x, col.group=NULL, ...)
```

Arguments

x	object of class limdil.
...	other arguments to be passed to plot or print. Note that pch and lty are reserved arguments for the plot method.
col.group	vector of colors for the groups of the same length as levels(x\$group).

Details

Produces a plot of a limiting dilution experiment similar to that in Bonnefoix et al. (2001). The basic design of the plot was made popular by Lefkovits and Waldmann (1979).

The plot shows frequencies and confidence intervals for the multiple groups. A novel feature is that assays with 100% successes are represented on the plot, by down-pointing triangles.

Author(s)

Yifang Hu and Gordon Smyth

References

Bonnefoix, T, Bonnefoix, P, Callanan, M, Verdiel, P, and Sotto, JJ (2001). Graphical representation of a generalized linear model-based statistical test estimating the fit of the single-hit poisson model to limiting dilution assays. *The Journal of Immunology* 167, 5725-5730.

Lefkovits, I, and Waldmann, H (1979). *Limiting dilution analysis of cells in the immune system*. Cambridge University Press, Cambridge.

See Also

[limdil](#) describes the limdil class.

power.fisher.test	<i>Power of Fisher's Exact Test for Comparing Proportions</i>
-------------------	---

Description

Calculate by simulation the power of Fisher's exact test for comparing two proportions given two margin counts.

Usage

```
power.fisher.test(p1, p2, n1, n2, alpha=0.05, nsim=100, alternative="two.sided")
```

Arguments

p1	first proportion to be compared.
p2	second proportion to be compared.
n1	first sample size.
n2	second sample size.
alpha	significance level.
nsim	number of data sets to simulate.
alternative	indicates the alternative hypothesis and must be one of "two.sided", "greater" or "less".

Details

Computes the power of Fisher's exact test for testing the null hypothesis that p_1 equals p_2 against the alternative that they are not equal.

Value

Estimated power of the test.

Author(s)

Gordon Smyth

See Also

[fisher.test](#), [power.t.test](#)

Examples

```
power.fisher.test(0.5,0.9,20,20) # 70% chance of detecting difference
```

qresiduals

Randomized Quantile Residuals

Description

Compute randomized quantile residuals for generalized linear models.

Usage

```
qresiduals(glm.obj,dispersion=NULL)
qresid(glm.obj,dispersion=NULL)
qres.binom(glm.obj)
qres.pois(glm.obj)
qres.nbinom(glm.obj)
qres.gamma(glm.obj,dispersion=NULL)
qres.invgauss(glm.obj,dispersion=NULL)
qres.tweedie(glm.obj,dispersion=NULL)
qres.default(glm.obj,dispersion=NULL)
```

Arguments

glm.obj	Object of class glm. The generalized linear model family is assumed to be binomial for qres.binom, poisson for qres.pois, negative binomial for qres.nbinom, Gamma for qres.gamma, inverse Gaussian for qres.invgauss or tweedie for qres.tweedie.
dispersion	a positive real number. Specifies the value of the dispersion parameter for a Gamma or inverse Gaussian generalized linear model if known. If NULL, the dispersion will be estimated by its Pearson estimator.

Details

Quantile residuals are based on the idea of inverting the estimated distribution function for each observation to obtain exactly standard normal residuals. In the case of discrete distributions, such as the binomial and Poisson, some randomization is introduced to produce continuous normal residuals. Quantile residuals are the residuals of choice for generalized linear models in large dispersion situations when the deviance and Pearson residuals can be grossly non-normal. Quantile residuals are the only useful residuals for binomial or Poisson data when the response takes on only a small number of distinct values.

Value

Numeric vector of standard normal quantile residuals.

Author(s)

Gordon Smyth

References

Dunn, K. P., and Smyth, G. K. (1996). Randomized quantile residuals. *Journal of Computational and Graphical Statistics* 5, 1-10. <http://www.statsci.org/smyth/pubs/residual.html>

See Also

[residuals.glm](#)

Examples

```
# Poisson example: quantile residuals show no granularity
y <- rpois(20,lambda=4)
x <- 1:20
fit <- glm(y~x, family=poisson)
qr <- qresiduals(fit)
qqnorm(qr)
abline(0,1)

# Gamma example:
# Quantile residuals are nearly normal while usual resids are not
y <- rchisq(20, df=1)
fit <- glm(y~1, family=Gamma)
qr <- qresiduals(fit, dispersion=2)
qqnorm(qr)
abline(0,1)

# Negative binomial example:
if(require("MASS")) {
fit <- glm(Days~Age,family=negative.binomial(2),data=quine)
summary(qresiduals(fit))
fit <- glm.nb(Days~Age,link=log,data = quine)
summary(qresiduals(fit))
}
```

remlscore

*REML for Heteroscedastic Regression***Description**

Fits a heteroscedastic regression model using residual maximum likelihood (REML).

Usage

```
remlscore(y, X, Z, trace=FALSE, tol=1e-5, maxit=40)
```

Arguments

y	numeric vector of responses
X	design matrix for predicting the mean
Z	design matrix for predicting the variance
trace	Logical variable. If true then output diagnostic information at each iteration.
tol	Convergence tolerance
maxit	Maximum number of iterations allowed

Details

Write $\mu_i = E(y_i)$ for the expectation of the i th response and $s_i = (y_i)$. We assume the heteroscedastic regression model

$$\begin{aligned}\mu_i &= \mathbf{x}_i^T \boldsymbol{\beta} \\ \log(\sigma_i^2) &= \mathbf{z}_i^T \boldsymbol{\gamma},\end{aligned}$$

where \mathbf{x}_i and \mathbf{z}_i are vectors of covariates, and $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ are vectors of regression coefficients affecting the mean and variance respectively.

Parameters are estimated by maximizing the REML likelihood using REML scoring as described in Smyth (2002).

Value

List with the following components:

beta	vector of regression coefficients for predicting the mean
se.beta	vector of standard errors for beta
gamma	vector of regression coefficients for predicting the variance
se.gam	vector of standard errors for gamma
mu	estimated means
phi	estimated variances
deviance	minus twice the REML log-likelihood
h	numeric vector of leverages

cov.beta	estimated covariance matrix for beta
cov.gam	estimated covariate matrix for gamma
iter	number of iterations used

Author(s)

Gordon Smyth

References

Smyth, G. K. (2002). An efficient algorithm for REML in heteroscedastic regression. *Journal of Computational and Graphical Statistics* **11**, 836-847.

Examples

```
data(welding)
attach(welding)
y <- Strength
# Reproduce results from Table 1 of Smyth (2002)
X <- cbind(1,(Drying+1)/2,(Material+1)/2)
colnames(X) <- c("1","B","C")
Z <- cbind(1,(Material+1)/2,(Method+1)/2,(Preheating+1)/2)
colnames(Z) <- c("1","C","H","I")
out <- remlscore(y,X,Z)
cbind(Estimate=out$gamma,SE=out$se.gam)
```

remlscoregamma	<i>Approximate REML for gamma regression with structured dispersion</i>
----------------	---

Description

Estimates structured dispersion effects using approximate REML with gamma responses.

Usage

```
remlscoregamma(y,X,Z,mlink="log",dlink="log",trace=FALSE,tol=1e-5,maxit=40)
```

Arguments

y	numeric vector of responses
X	design matrix for predicting the mean
Z	design matrix for predicting the variance
mlink	character string or numeric value specifying link for mean model
dlink	character string or numeric value specifying link for dispersion model
trace	Logical variable. If true then output diagnostic information at each iteration.
tol	Convergence tolerance
maxit	Maximum number of iterations allowed

Details

Write $\mu_i = E(y_i)$ for the expectation of the i th response and $s_i = (y_i)$. We assume the heteroscedastic regression model

$$\begin{aligned}\mu_i &= \mathbf{x}_i^T \boldsymbol{\beta} \\ \log(\sigma_i^2) &= \mathbf{z}_i^T \boldsymbol{\gamma},\end{aligned}$$

where \mathbf{x}_i and \mathbf{z}_i are vectors of covariates, and $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ are vectors of regression coefficients affecting the mean and variance respectively.

Parameters are estimated by maximizing the REML likelihood using REML scoring as described in Smyth and Verbyla (2001). See also Smyth and Verbyla (1999a,b).

Value

List with the following components:

beta	Vector of regression coefficients for predicting the mean
se.beta	<Standard errors for beta
gamma	Vector of regression coefficients for predicting the variance
se.gam	Standard errors for gamma
mu	Estimated means
phi	Estimated dispersions
deviance	Minus twice the REML log-likelihood
h	Leverages

References

Smyth, G. K., and Verbyla, A. P. (1999a). Adjusted likelihood methods for modelling dispersion in generalized linear models. *Environmetrics* 10, 695-709. <http://www.statsci.org/smyth/pubs/earlier.html>

Smyth, G. K., and Verbyla, A. P. (1999b). Double generalized linear models: approximate REML and diagnostics. In *Statistical Modelling: Proceedings of the 14th International Workshop on Statistical Modelling*, Graz, Austria, July 19-23, 1999, H. Friedl, A. Berghold, G. Kauermann (eds.), Technical University, Graz, Austria, pages 66-80. <http://www.statsci.org/smyth/pubs/earlier.html>

Smyth, G. K., and Verbyla, A. P. (2001). Leverage adjustments for dispersion modelling in generalized nonlinear models. Unpublished technical report. <http://www.statsci.org/smyth/pubs/dglm.ps>

Examples

```
data(welding)
attach(welding)
y <- Strength
X <- cbind(1,(Drying+1)/2,(Material+1)/2)
colnames(X) <- c("1","B","C")
Z <- cbind(1,(Material+1)/2,(Method+1)/2,(Preheating+1)/2)
colnames(Z) <- c("1","C","H","I")
out <- remlscoregamma(y,X,Z)
```

Description

This function is kept here so as not to break code that depends on it, but has been replaced by `binomTest` in the edgeR Bioconductor package and is no longer updated. It may be removed in a later release of this package.

Computes p-values for differential abundance for each tag between two digital libraries, conditioning on the total count for each tag. The counts in each group as a proportion of the whole are assumed to follow a binomial distribution.

Usage

```
sage.test(x, y, n1=sum(x), n2=sum(y))
```

Arguments

x	integer vector giving counts in first library. Non-integer values are rounded to the nearest integer.
y	integer vector giving counts in second library. Non-integer values are rounded to the nearest integer.
n1	total number of tags in first library. Non-integer values are rounded to the nearest integer.
n2	total number of tags in second library. Non-integer values are rounded to the nearest integer.

Details

This function was originally written for comparing SAGE libraries (a method for counting the frequency of sequence tags in samples of RNA). It can however be used for comparing any two digital libraries from RNA-Seq, ChIP-Seq or other technologies with respect to technical variation.

An exact two-sided binomial test is computed for each tag. This test is closely related to Fisher's exact test for 2x2 contingency tables but, unlike Fisher's test, it conditions on the total number of counts for each tag. The null hypothesis is that the expected counts are in the same proportions as the library sizes, i.e., that the binomial probability for the first library is $n1/(n1+n2)$.

The two-sided rejection region is chosen analogously to Fisher's test. Specifically, the rejection region consists of those values with smallest probabilities under the null hypothesis.

When the counts are reasonably large, the binomial test, Fisher's test and Pearson's chisquare all give the same results. When the counts are smaller, the binomial test is usually to be preferred in this context.

This function is a later version of the earlier `sage.test` function in the `sagenhaft` Bioconductor package. This function has been replaced by `binomTest` in the edgeR package.

Value

Numeric vector of p-values.

Author(s)

Gordon Smyth

References

http://en.wikipedia.org/wiki/Binomial_test

http://en.wikipedia.org/wiki/Fisher's_exact_test

http://en.wikipedia.org/wiki/Serial_analysis_of_gene_expression

<http://en.wikipedia.org/wiki/RNA-Seq>

See Also

[binomTest](#) (edgeR package), [binom.test](#) (stats package)

Examples

```
sage.test(c(0,5,10),c(0,30,50),n1=10000,n2=15000)
# Univariate equivalents:
binom.test(5,5+30,p=10000/(10000+15000))$p.value
binom.test(10,10+50,p=10000/(10000+15000))$p.value
```

tweedie

Tweedie Generalized Linear Models

Description

Produces a generalized linear model family object with any power variance function and any power link. Includes the Gaussian, Poisson, gamma and inverse-Gaussian families as special cases.

Usage

```
tweedie(var.power=0, link.power=1-var.power)
```

Arguments

`var.power` index of power variance function

`link.power` index of power link function. `link.power=0` produces a log-link. Defaults to the canonical link, which is `1-var.power`.

Details

This function provides access to a range of generalized linear model response distributions which are not otherwise provided by R, or any other package for that matter. It is also useful for accessing distribution/link combinations which are disallowed by the R `glm` function.

Let $\mu_i = E(y_i)$ be the expectation of the i th response. We assume that

$$\mu_i^q = x_i^T b, \text{var}(y_i) = \phi \mu_i^p$$

where x_i is a vector of covariates and b is a vector of regression coefficients, for some ϕ , p and q . This family is specified by `var.power = p` and `link.power = q`. A value of zero for q is interpreted as $\log(\mu_i) = x_i^T b$.

The variance power p characterizes the distribution of the responses y . The following are some special cases:

p	Response distribution
0	Normal
1	Poisson
(1, 2)	Compound Poisson, non-negative with mass at zero
2	Gamma
3	Inverse-Gaussian
> 2	Stable, with support on the positive reals

The name Tweedie has been associated with this family by Joergensen (1987) in honour of M. C. K. Tweedie.

Value

A family object, which is a list of functions and expressions used by `glm` and `gam` in their iteratively reweighted least-squares algorithms. See `family` and `glm` in the R base help for details.

Author(s)

Gordon Smyth

References

- Tweedie, M. C. K. (1984). An index which distinguishes between some important exponential families. In *Statistics: Applications and New Directions*. Proceedings of the Indian Statistical Institute Golden Jubilee International Conference. (Eds. J. K. Ghosh and J. Roy), pp. 579-604. Calcutta: Indian Statistical Institute.
- Joergensen, B. (1987). Exponential dispersion models. *J. R. Statist. Soc. B* **49**, 127-162.
- Smyth, G. K. (1996). Regression modelling of quantity data with exact zeroes. Proceedings of the Second Australia-Japan Workshop on Stochastic Models in Engineering, Technology and Management. Technology Management Centre, University of Queensland, pp. 572-580.
- Joergensen, B. (1997). *Theory of Dispersion Models*, Chapman and Hall, London.
- Smyth, G. K., and Verbyla, A. P., (1999). Adjusted likelihood methods for modelling dispersion in generalized linear models. *Environmetrics* **10**, 695-709.

See Also

[glm](#), [family](#), [dtweedie](#)

Examples

```
y <- rgamma(20,shape=5)
x <- 1:20
# Fit a poisson generalized linear model with identity link
glm(y~x,family=tweedie(var.power=1,link.power=1))

# Fit an inverse-Gaussian glm with log-link
glm(y~x,family=tweedie(var.power=3,link.power=0))
```

welding

Data: Tensile Strength of Welds

Description

This is a highly fractionated two-level factorial design employed as a screening design in an off-line welding experiment performed by the National Railway Corporation of Japan. There were 16 runs and 9 experimental factors. The response variable is the observed tensile strength of the weld, one of several quality characteristics measured. All other variables are at plus and minus levels.

Usage

```
data(welding)
```

Format

A data frame containing the following variables. All the explanatory variables are numeric with two levels, -1 and 1.

Variable	Description
Rods	Kind of welding rods
Drying	Period of drying
Material	Welded material
Thickness	Thickness
Angle	Angle
Opening	Opening
Current	Current
Method	Welding method
Preheating	Preheating
Strength	Tensile strength of the weld in kg/mm. The response variable.

Source

<http://www.statsci.org/data/general/welding.html>

References

- Smyth, G. K., Huele, F., and Verbyla, A. P. (2001). Exact and approximate REML for heteroscedastic regression. *Statistical Modelling* **1**, 161-175.
- Smyth, G. K. (2002). An efficient algorithm for REML in heteroscedastic regression. *Journal of Computational and Graphical Statistics* **11**, 1-12.

Index

- *Topic **algebra**
 - matvec, 20
- *Topic **array**
 - matvec, 20
- *Topic **datasets**
 - welding, 36
- *Topic **distribution**
 - invgauss, 18
- *Topic **documentation**
 - 1.StatMod, 2
- *Topic **htest**
 - hommel.test, 17
 - permp, 25
 - power.fisher.test, 27
 - sage.test, 33
- *Topic **math**
 - gauss.quad, 10
 - gauss.quad.prob, 11
 - logmdigamma, 19
- *Topic **models**
 - Digamma, 4
- *Topic **regression**
 - deprecated, 3
 - elda, 6
 - fitNBP, 8
 - glm.scoretest, 13
 - glmgam.fit, 14
 - growthcurve, 15
 - meanT, 21
 - mixedModel2, 22
 - plot.limdil, 26
 - qresiduals, 28
 - remlscore, 30
 - remlscoregamma, 31
 - tweedie, 34
- 1.StatMod, 2
- add1, 14
- binom.test, 34
- binomTest, 33, 34
- canonic.digamma (Digamma), 4
- compareGrowthCurves, 17, 21
- compareGrowthCurves (growthcurve), 15
- compareTwoGrowthCurves, 17, 21
- compareTwoGrowthCurves (growthcurve), 15
- cumulant.digamma (Digamma), 4
- d2cumulant.digamma (Digamma), 4
- deprecated, 3
- Digamma, 4
- digamma, 20
- dinvgauss (invgauss), 18
- dtweedie, 36
- elda, 6
- family, 36
- fisher.test, 28
- fitNBP, 8
- gauss.quad, 10, 12
- gauss.quad.prob, 11, 11
- glm, 13, 14, 36
- glm.scoretest, 13
- glmgam.fit, 14, 23
- glmnb.fit (glmgam.fit), 14
- growthcurve, 15
- hommel.test, 17
- integrate, 11, 12
- InverseGaussian (invgauss), 18
- invgauss, 18
- limdil, 2, 27
- limdil (elda), 6
- lm, 23
- lm.fit, 23
- lme, 23

logmdigamma, 19

make.link, 5

matvec, 20

meanT, 21

meanval.digamma (Digamma), 4

mixedModel2, 4, 22

mixedModel2Fit, 4, 15

mixedModel2Fit (mixedModel2), 22

mscale, 24

p.adjust, 18

permp, 25

pinvgauss (invgauss), 18

plot, 7

plot.limdil, 26

plotGrowthCurves (growthcurve), 15

power.fisher.test, 27

power.t.test, 28

print, 7

print.limdil (plot.limdil), 26

qinvgauss (invgauss), 18

qres.binom (qresiduals), 28

qres.default (qresiduals), 28

qres.gamma (qresiduals), 28

qres.invgauss (qresiduals), 28

qres.nbinom (qresiduals), 28

qres.pois (qresiduals), 28

qres.tweedie (qresiduals), 28

qresid (qresiduals), 28

qresiduals, 28

quasi, 5

randomizedBlock (deprecated), 3

randomizedBlockFit (deprecated), 3

remlscore, 30

remlscoregamma, 31

residuals.glm, 29

rinvgauss (invgauss), 18

sage.test, 9, 33

statmod-deprecated (deprecated), 3

summary.glm, 13

tweedie, 34

unitdeviance.digamma (Digamma), 4

varfun.digamma (Digamma), 4

vecmat (matvec), 20

welding, 36