

# Package ‘multinomRob’

April 17, 2009

**Version** 1.8-2

**Date** 2006/02/09

**Title** Robust Estimation of Overdispersed Multinomial Regression Models

**Author** Walter R. Mebane, Jr. <wrm1@cornell.edu>, Jasjeet Singh Sekhon <sekhon@berkeley.edu>

**Maintainer** Jasjeet Singh Sekhon <sekhon@berkeley.edu>

**Description** MNL and overdispersed multinomial regression using robust (LQD and tanh) estimation

**Depends** R (>= 1.7.1), rgenoud (>= 2.9), MASS (>= 7.1-8), mvtnorm (>= 0.6-3)

**License** GPL (>= 2)

**URL** <http://sekhon.polisci.berkeley.edu/>

**Repository** CRAN

**Date/Publication** 2007-02-13 11:11:38

## R topics documented:

Multinomial Multivariate-T Regression . . . . .	2
Multinomial Regression . . . . .	3
Multinomial Regression Tanh Estimator . . . . .	6
Multinomial Regression Tanh Estimator Gauss-Newton Optimization . . . . .	8
rmultinomial . . . . .	10
Robust Multinomial Regression . . . . .	11
<b>Index</b>	<b>17</b>

---

Multinomial Multivariate-T Regression  
*Multinomial Multivariate-T Estimation*

---

**Description**

`multinomT` fits the multinomial multivariate-t regression for grouped count data. This function is not meant to be called directly by the user. It is called by `multinomRob`, which constructs the various arguments.

**Usage**

```
multinomT(Yp, Xarray, xvec, jacstack, start = NA, nobsvec, fixed.df = NA)
```

**Arguments**

<code>Yp</code>	Matrix (observations by alternatives) of outcome proportions. Values must be between 0 and 1. Missing data (NA values) are not allowed.
<code>Xarray</code>	Array of regressors. <code>dim(Xarray) = c(observations, parameters, alternatives)</code> .
<code>xvec</code>	Matrix (parameters by alternatives) that represents the model structure. It has a 1 for an estimated parameter, an integer greater than 1 for an estimated parameter constrained equal to another estimated parameter (all parameters constrained to be equal to one another have the same integer value in <code>xvec</code> ) and a 0 otherwise.
<code>jacstack</code>	Array of regressors used to facilitate computing the gradient and the hessian matrix. <code>dim(jacstack) = c(observations, unique parameters, alternatives)</code> .
<code>start</code>	A list of starting values of three kinds of parameters: <code>start\$beta</code> , the values for the regression coefficients; <code>start\$Omega</code> , the values for the variance-covariance matrix; <code>start\$df</code> , the value for the multivariate-t degrees of freedom parameter.
<code>nobsvec</code>	Vector of the total number of counts for each observation.
<code>fixed.df</code>	The degrees of freedom to be used for the multivariate-t distribution. When this is specified, the DF will not be estimated.

**Details**

The function often provides good starting values for `multinomRob`'s LQD estimator, but the standard errors it reports are not correct, in part because they ignore heteroscedasticity.

**Value**

<code>call</code>	Names and values of all of the arguments which were passed to the function. See <code>match.call</code> for further details.
<code>logL</code>	Log likelihood.
<code>deviance</code>	Deviance.

<code>par</code>	A list of three kinds of parameter estimates: <code>par\$beta</code> , the estimates for the regression coefficients; <code>par\$Omega</code> , the estimates for the variance-covariance matrix; <code>par\$df</code> , the estimate of the multivariate-t degrees of freedom parameter.
<code>se</code>	Vector of standard errors for the regression coefficients. WARNING: these are not correct in part because the model ignores heteroscedasticity.
<code>optim</code>	Returned by <code>optim</code> .
<code>pred</code>	A matrix of predicted probabilities with the same dimensions as $Y_p$ .

**Author(s)**

Walter R. Mebane, Jr., Cornell University, [wrm1@cornell.edu](mailto:wrm1@cornell.edu), <http://macht.arts.cornell.edu/wrm1/>

Jasjeet S. Sekhon, UC Berkeley, [sekhon@berkeley.edu](mailto:sekhon@berkeley.edu), <http://sekhon.berkeley.edu/>

**References**

Walter R. Mebane, Jr. and Jasjeet Singh Sekhon. 2004. "Robust Estimation and Outlier Detection for Overdispersed Multinomial Models of Count Data." *American Journal of Political Science* 48 (April): 391–410. <http://sekhon.berkeley.edu/multinom.pdf>

For additional documentation please visit <http://sekhon.berkeley.edu/robust/>.

**See Also**

[match.call.optim](#).

---

Multinomial Regression

*Multinomial Regression Maximum Likelihood Estimator with Overdispersion*

---

**Description**

`multinomMLE` estimates the coefficients of the multinomial regression model for grouped count data by maximum likelihood, then computes a moment estimator for overdispersion and reports standard errors for the coefficients that take overdispersion into account. This function is not meant to be called directly by the user. It is called by `multinomRob`, which constructs the various arguments.

**Usage**

```
multinomMLE(Y, Ypos, Xarray, xvec, jacstack, itmax=100, xvar.labels,
            choice.labels, MLEonly=FALSE, print.level=0)
```

**Arguments**

<code>Y</code>	Matrix (observations by alternatives) of outcome counts. Values must be non-negative. Missing data (NA values) are not allowed.
<code>Ypos</code>	Matrix indicating which elements of <code>Y</code> are counts to be analyzed (TRUE) and which are values to be skipped (FALSE). This allows the set of outcome alternatives to vary over observations.
<code>Xarray</code>	Array of regressors. $\text{dim}(Xarray) = c(\text{observations}, \text{parameters}, \text{alternatives})$ .
<code>xvec</code>	Matrix (parameters by alternatives) that represents the model structure. It has a 1 for an estimated parameter, an integer greater than 1 for an estimated parameter constrained equal to another estimated parameter (all parameters constrained to be equal to one another have the same integer value in <code>xvec</code> ) and a 0 otherwise.
<code>jacstack</code>	Array of regressors used to facilitate computing the gradient and the hessian matrix. $\text{dim}(jacstack) = c(\text{observations}, \text{unique parameters}, \text{alternatives})$ .
<code>itmax</code>	The maximum number of iterations to be done in the Gauss-Newton optimization.
<code>xvar.labels</code>	Vector of labels for observations.
<code>choice.labels</code>	Vector of labels for outcome alternatives.
<code>MLEonly</code>	If TRUE, then only the standard maximum-likelihood MNL model is estimated—i.e., no overdispersion parameter is estimated.
<code>print.level</code>	Specify 0 for minimal printing (error messages only) or 3 to print details about the MLE computations.

**Details**

Following the generalized linear models approach, the coefficient parameters in an overdispersed multinomial regression model may be estimated using the likelihood for a standard multinomial regression model. A moment estimator may be used for the dispersion parameter, given the coefficient estimates, with little efficiency loss.

**Value**

`multinomMLE` returns a list containing the following objects. The returned objects are:

<code>coefficients</code>	The maximum likelihood coefficient estimates in matrix format. The value 0 is used in the matrix to fill in for values that do not correspond to a regressor.
<code>coeffvec</code>	A vector containing the maximum likelihood coefficient estimates.
<code>dispersion</code>	Moment estimate of the dispersion: mean sum of squared orthogonalized residuals (adjusted for degrees of freedom lost to estimated coefficients).
<code>se</code>	The MLE coefficient estimate standard errors derived from the asymptotic covariance estimated using the Hessian matrix (observed information).
<code>se.opg</code>	The MLE coefficient estimate standard errors derived from the asymptotic covariance estimated using the outer product of the gradient (expected information) divided by the moment estimate of the dispersion. Not provided if <code>MLEonly==TRUE</code> .

<code>se.hes</code>	The MLE coefficient estimate standard errors derived from the asymptotic covariance estimated using the Hessian matrix (observed information). Same as <code>se</code> ; included for backward compatibility.
<code>se.sw</code>	The MLE coefficient estimate standard errors derived from the asymptotic covariance estimated using the estimated asymptotic sandwich covariance estimate. Not provided if <code>MLEonly==TRUE</code> .
<code>se.vec</code>	<code>se</code> in vector form.
<code>se.opg.vec</code>	<code>se.opg</code> in vector form.
<code>se.hes.vec</code>	<code>se.hes</code> in vector form.
<code>se.sw.vec</code>	<code>se.sw</code> in vector form.
<code>A</code>	The outer product of the gradient (expected information) divided by the moment estimate of the dispersion.
<code>B</code>	The inverse of the hessian matrix (observed formation).
<code>covmat</code>	Sandwich estimate of the asymptotic covariance of the maximum likelihood coefficient estimates.
<code>iters</code>	Number of Gauss-Newton iterations.
<code>error</code>	Exit error code.
<code>GNlist</code>	List reporting final results of the Gauss-Newton optimization. Elements: <code>coefficients</code> , vector of coefficient parameters (same as <code>coeffvec</code> value in list returned by <code>multinomMLE</code> ); <code>tvec</code> , matrix of coefficient parameters (same as <code>coefficients</code> value in list returned by <code>multinomMLE</code> ); <code>formation</code> , inverse Hessian matrix; <code>score</code> , score (or gradient element) matrix; <code>LLvals</code> , list containing log-likelihood value; <code>convflag</code> , TRUE/FALSE convergence flag; <code>iters</code> , number of iterations done in final Gauss-Newton stage; <code>posdef</code> , TRUE if Hessian is positive definite.
<code>sigma2</code>	Moment estimate of the dispersion: mean sum of squared orthogonalized residuals (adjusted for degrees of freedom lost to estimated coefficients).
<code>Y</code>	The same <code>Y</code> matrix that was supplied as input, except modified by having done <code>Y[!Ypos] &lt;- 0</code> .
<code>Ypos</code>	The same <code>Ypos</code> matrix that was supplied as input.
<code>fitted.prob</code>	The matrix of predicted probabilities for each category for each observation based on the coefficient estimates.
<code>jacstack</code>	The same <code>jacstack</code> that was supplied as an input argument.

**Author(s)**

Walter R. Mebane, Jr., Cornell University, [wrm1@cornell.edu](mailto:wrm1@cornell.edu), <http://macht.arts.cornell.edu/wrm1/>

Jasjeet S. Sekhon, UC Berkeley, [sekhon@berkeley.edu](mailto:sekhon@berkeley.edu), <http://sekhon.berkeley.edu/>

## References

Walter R. Mebane, Jr. and Jasjeet Singh Sekhon. 2004. “Robust Estimation and Outlier Detection for Overdispersed Multinomial Models of Count Data.” *American Journal of Political Science* 48 (April): 391–410. <http://sekhon.berkeley.edu/multinom.pdf>

For additional documentation please visit <http://sekhon.berkeley.edu/robust/>.

---

Multinomial Regression Tanh Estimator

*Multinomial Regression Hyperbolic Tangent (Tanh) Estimator*

---

## Description

`multinomTanh` fits the overdispersed multinomial regression model for grouped count data using the hyperbolic tangent (tanh) estimator. This function is not meant to be called directly by the user. It is called by `multinomRob`, which constructs the various arguments.

## Usage

```
multinomTanh(Y, Ypos, X, jacstack, xvec, tvec, pop, s2,
             xvar.labels, choice.labels, print.level = 0)
```

## Arguments

<code>Y</code>	Matrix (observations by alternatives) of outcome counts. Values must be non-negative. Missing data (NA values) are not allowed.
<code>Ypos</code>	Matrix indicating which elements of <code>Y</code> are counts to be analyzed (TRUE) and which are values to be skipped (FALSE). This allows the set of outcome alternatives to vary over observations.
<code>X</code>	Array of regressors. $\dim(X) = c(\text{observations}, \text{parameters}, \text{alternatives})$ .
<code>jacstack</code>	Array of regressors used to facilitate computing the gradient and the hessian matrix. $\dim(\text{jacstack}) = c(\text{observations}, \text{unique parameters}, \text{alternatives})$ .
<code>xvec</code>	Matrix (parameters by alternatives) that represents the model structure. It has a 1 for an estimated parameter, an integer greater than 1 for an estimated parameter constrained equal to another estimated parameter (all parameters constrained to be equal to one another have the same integer value in <code>xvec</code> ) and a 0 otherwise.
<code>tvec</code>	Starting values for the regression coefficient parameters, as a matrix (parameters by alternatives). Parameters that are involved in equality constraints are repeated in <code>tvec</code> .
<code>pop</code>	Vector giving the total number of counts for each observation. In general, <code>pop &lt;- apply(Y * ifelse(Ypos, 1, 0), 1, sum)</code> .
<code>s2</code>	Overdispersion value. In <code>multinomRob</code> this is the square of the LQD scale estimate.
<code>xvar.labels</code>	Vector of labels for observations.

<code>choice.labels</code>	Vector of labels for outcome alternatives.
<code>print.level</code>	Specify 0 for minimal printing (error messages only) or 2 to print details about the tanh computations.

## Details

The tanh estimator is a redescending M-estimator. Given an estimate of the scale of the overdispersion, the tanh estimator estimates the coefficient parameters of the linear predictors of the multinomial regression model.

## Value

`multinomTanh` returns a list of 5 objects. The returned objects are:

<code>mtanh</code>	List of tanh estimation results from function <code>mGNTanh</code> .
<code>weights</code>	The matrix of tanh weights for the orthogonalized residuals. The matrix has the same dimensions as the outcome count matrix $Y$ . The first column of the matrix has names for the observations, and the remaining columns contain the weights. Each of the latter columns has a name derived from the <code>choice.labels</code> vector: column $i+1$ is named <code>paste("weights:", choice.labels[i], sep="")</code> . If <code>sum(Ypos[i,]==FALSE)&gt;0</code> , then values of NA appear in <code>weights[i,]</code> , with <code>sum(is.na(weights[i,]))==sum(!Ypos[i,])</code> . The NA values will be the last values in the affected row of the <code>weights</code> matrix, regardless of which outcome alternatives were unavailable for the observation.
<code>Hdiag</code>	The matrix of weights used to fully studentize the orthogonalized residuals. The matrix has the same dimensions as the outcome count matrix $Y$ . The first column of the matrix has names for the observations, and the remaining columns contain the weights. Each of the latter columns has a name derived from the <code>choice.labels</code> vector: column $i+1$ is named <code>paste("Hdiag:", choice.labels[i], sep="")</code> . If <code>sum(Ypos[i,]==FALSE)&gt;0</code> , then values of 0 appear in <code>Hdiag[i,]</code> , with <code>sum(is.na(Hdiag[i,]))==sum(!Ypos[i,])</code> . The 0 values created for this reason will be the last values in the affected row of the <code>Hdiag</code> matrix, regardless of which outcome alternatives were unavailable for the observation.
<code>cr</code>	List of predicted outcome counts, studentized residuals and standardized residuals.
<code>tvec</code>	The tanh coefficient estimates in matrix format. The matrix has one column for each outcome alternative. The label for each row of the matrix gives the names of the regressors to which the coefficient values in the row apply. The regressor names in each label are separated by a forward slash (/), and NA is used to denote that no regressor is associated with the corresponding value in the matrix. The value 0 is used in the matrix to fill in for values that do not correspond to a regressor.

**Author(s)**

Walter R. Mebane, Jr., Cornell University, [wrm1@cornell.edu](mailto:wrm1@cornell.edu), <http://macht.arts.cornell.edu/wrm1/>

Jasjeet S. Sekhon, UC Berkeley, [sekhon@berkeley.edu](mailto:sekhon@berkeley.edu), <http://sekhon.berkeley.edu/>

**References**

Walter R. Mebane, Jr. and Jasjeet Singh Sekhon. 2004. "Robust Estimation and Outlier Detection for Overdispersed Multinomial Models of Count Data." *American Journal of Political Science* 48 (April): 391–410. <http://sekhon.berkeley.edu/multinom.pdf>

For additional documentation please visit <http://sekhon.berkeley.edu/robust/>.

---

Multinomial Regression Tanh Estimator Gauss-Newton Optimization  
*Multinomial Regression Hyperbolic Tangent (Tanh) Estimator  
 Gauss-Newton Optimization*

---

**Description**

`mGNtanh` uses Gauss-Newton optimization to compute the hyperbolic tangent (tanh) estimator for the overdispersed multinomial regression model for grouped count data. This function is not meant to be called directly by the user. It is called by `multinomRob`, which constructs the various arguments.

**Usage**

```
mGNtanh(bstart, sigma2, resstart, Y, Ypos, Xarray, xvec, tvec,
        jacstack, itmax = 100, print.level = 0)
```

**Arguments**

<code>bstart</code>	Vector of starting values for the coefficient parameters.
<code>sigma2</code>	Value of the dispersion parameter (variance). The estimator does not update this value.
<code>resstart</code>	Array of initial orthogonalized (but not standardized) residuals.
<code>Y</code>	Matrix (observations by alternatives) of outcome counts. Values must be non-negative. Missing data (NA values) are not allowed.
<code>Ypos</code>	Matrix indicating which elements of <code>Y</code> are counts to be analyzed (TRUE) and which are values to be skipped (FALSE). This allows the set of outcome alternatives to vary over observations.
<code>Xarray</code>	Array of regressors. $\dim(Xarray) = c(\text{observations}, \text{parameters}, \text{alternatives})$ .

<code>xvec</code>	Matrix (parameters by alternatives) that represents the model structure. It has a 1 for an estimated parameter, an integer greater than 1 for an estimated parameter constrained equal to another estimated parameter (all parameters constrained to be equal to one another have the same integer value in <code>xvec</code> ) and a 0 otherwise.
<code>tvec</code>	Starting values for the regression coefficient parameters, as a matrix (parameters by alternatives). Parameters that are involved in equality constraints are repeated in <code>tvec</code> .
<code>jacstack</code>	Array of regressors used to facilitate computing the gradient and the hessian matrix. $\dim(\text{jacstack}) = c(\text{observations}, \text{unique parameters}, \text{alternatives})$ .
<code>itmax</code>	Maximum number of Gauss-Newton stages. Each stage does at most 100 Gauss-Newton steps.
<code>print.level</code>	Specify 0 for minimal printing (error messages only) or 2 to print details about the tanh computations.

### Details

The tanh estimator is a redescending M-estimator. Given an estimate of the scale of the overdispersion, the tanh estimator estimates the coefficient parameters of the linear predictors of the multinomial regression model.

### Value

`mGNtanh` returns a list of 16 objects. The returned objects are:

<code>coefficients</code>	The tanh coefficient estimates in matrix format. The matrix has one column for each outcome alternative. The label for each row of the matrix gives the names of the regressors to which the coefficient values in the row apply. The regressor names in each label are separated by a forward slash (/), and NA is used to denote that no regressor is associated with the corresponding value in the matrix. The value 0 is used in the matrix to fill in for values that do not correspond to a regressor.
<code>coeffvec</code>	A vector containing the tanh coefficient estimates.
<code>dispersion</code>	Value of the dispersion parameter (variance). This is the value specified in the argument <code>sigma2</code> in the call to the function.
<code>w</code>	Vector of weights based on the tanh estimator's <code>psi</code> function for each observation.
<code>psi</code>	Vector of values of the tanh estimator's <code>psi</code> function for each observation.
<code>A</code>	The outer product of the gradient (expected information) divided by the moment estimate of the dispersion.
<code>B</code>	The inverse of the hessian matrix (observed formation).
<code>covmat</code>	Sandwich estimate of the asymptotic covariance of the tanh coefficient estimates.
<code>iters</code>	Number of Gauss-Newton iterations.
<code>error</code>	Error code: 0, no errors; 2, $\sum(w) < \text{nobs} * (\text{ncats} - 1) / 2$ (weights are too small); 32, Hessian not positive definite in the final Newton step.

<code>GNlist</code>	List reporting final results of the Gauss-Newton optimization. Elements: <code>coefficients</code> , vector of coefficient parameters (same as <code>coeffvec</code> value in list returned by <code>mGNtanh</code> ); <code>tvec</code> , matrix of coefficient parameters (same as <code>coefficients</code> value in list returned by <code>mGNtanh</code> ); <code>formation</code> , inverse Hessian matrix; <code>score</code> , score (or gradient element) matrix; <code>LLvals</code> , list containing weighted ( <code>LLvals\$LL</code> ) and unweighted ( <code>LLvals\$LLu</code> ) log-likelihood values; <code>convflag</code> , TRUE/FALSE convergence flag; <code>iters</code> , number of iterations done in final Gauss-Newton stage; <code>posdef</code> , TRUE if Hessian is positive definite.
<code>tanhsigma2</code>	The tanh overdispersion parameter estimate, which is a weighted moment estimate of the dispersion: weighted mean sum of squared orthogonalized residuals (adjusted for effective sample size after weighting and degrees of freedom lost to estimated coefficients).
<code>Y</code>	The same <code>Y</code> matrix that was supplied as input, except modified by having done <code>Y[!Ypos] &lt;- 0</code> .
<code>Ypos</code>	The same <code>Ypos</code> matrix that was supplied as input.
<code>probrmat</code>	The matrix of predicted probabilities for each category for each observation based on the coefficient estimates.
<code>jacstack</code>	The same <code>jacstack</code> that was supplied as an input argument.
<code>Xarray</code>	The same <code>Xarray</code> that was supplied as an input argument.

**Author(s)**

Walter R. Mebane, Jr., Cornell University, [wrm1@cornell.edu](mailto:wrm1@cornell.edu), <http://macht.arts.cornell.edu/wrm1/>

Jasjeet S. Sekhon, UC Berkeley, [sekhon@berkeley.edu](mailto:sekhon@berkeley.edu), <http://sekhon.berkeley.edu/>

**References**

Walter R. Mebane, Jr. and Jasjeet Singh Sekhon. 2004. "Robust Estimation and Outlier Detection for Overdispersed Multinomial Models of Count Data." *American Journal of Political Science* 48 (April): 391–410. <http://sekhon.berkeley.edu/multinom.pdf>

For additional documentation please visit <http://sekhon.berkeley.edu/robust/>.

---

`rmultinomial`

*Random Number Generator for the Multinomial Distribution*

---

**Description**

Generates a random count vector for one observation of a multinomial distribution for `n` trials with probability vector `pr`.

**Usage**

```
rmultinomial(n = 5, pr = c(0.5, 0.5), long = FALSE)
```

**Arguments**

<code>n</code>	Number of trials.
<code>pr</code>	Probability vector.
<code>long</code>	TRUE to choose one generator, FALSE to choose another one.

**Details**

Generates a random count vector for one observation of a multinomial distribution for `n` trials with probability vector `pr`.

**Value**

<code>x</code>	Vector of counts.
----------------	-------------------

**Note**

This function is only used in the examples and not in the `multinomRob` code.

**Author(s)**

Walter R. Mebane, Jr., Cornell University, [⟨wrm1@cornell.edu⟩](mailto:wrm1@cornell.edu), <http://macht.arts.cornell.edu/wrm1/>

Jasjeet S. Sekhon, UC Berkeley, [⟨sekhon@berkeley.edu⟩](mailto:sekhon@berkeley.edu), <http://sekhon.berkeley.edu/>

**Examples**

```
rmultinomial(10, c(.3, .3, .4));
```

---

Robust Multinomial Regression  
*Multinomial Robust Estimation*

---

**Description**

`multinomRob` fits the overdispersed multinomial regression model for grouped count data using the hyperbolic tangent (`tanh`) and least quartile difference (LQD) robust estimators.

**Usage**

```
multinomRob(model, data, starting.values=NULL, equality=NULL,  
            genoud.parms=NULL, print.level=0, iter = FALSE,  
            maxiter = 10, multinom.t=1, multinom.t.df=NA,  
            MLEonly=FALSE)
```

**Arguments**

<code>model</code>	<p>The regression model specification. This is a list of formulas, with one formula for each category of outcomes for which counts have been measured for each observation. For example, in the following,</p> <pre>model=list(y1 ~ x1, y2 ~ x2, y3 ~ 0)</pre> <p>the outcome variables containing counts are <code>y1</code>, <code>y2</code> and <code>y3</code>, and the linear predictor for <code>y1</code> is a coefficient times <code>x1</code> plus a constant, the linear predictor for <code>y2</code> is a coefficient times <code>x2</code> plus a constant, and the linear predictor for <code>y3</code> is zero. Each formula has the format <code>countvar ~ RHS</code>, where <code>countvar</code> is the name of a vector, in the dataframe referenced by the <code>data</code> argument, that gives the counts for all observations for one category. <code>RHS</code> denotes the right-hand side of a formula using the usual syntax for formulas, where each variable in the formula is the name of a vector in the dataframe referenced by the <code>data</code> argument. For example, a <code>RHS</code> specification of <code>var1 + var2*var3</code> would specify that the regressors are to be <code>var1</code>, <code>var2</code>, <code>var3</code>, the terms generated by the interaction <code>var2:var3</code>, and the constant.</p> <p>The set of outcome alternatives may be specified to vary over observations, by putting in a negative value for alternatives that do not exist for particular observations. If the value of an outcome variable is negative for an observation, then that outcome is considered not available for that observation. The predicted counts for that observation are defined only for the available observations and are based on the linear predictors for the available observations. The same set of coefficient parameter values are used for all observations. Any observation for which fewer than two outcomes are available is omitted.</p> <p>Observations with missing data (NA) in any outcome variable or regressor are omitted (listwise deletion).</p> <p>In a model that has the same regressors for every category, except for one category for which there are no regressors in order to identify the model (the reference category), the <code>RHS</code> specification must be given for all the categories except the reference category. The formula for the reference category must include a <code>RHS</code> specification that explicitly omits the constant, e.g., <code>countvar ~ -1</code> or <code>countvar ~ 0</code>. The number of coefficient parameters to be estimated equals the number of terms generated by all the formulas, subject to equality constraints that may be specified using the <code>equality</code> argument.</p>
<code>data</code>	The dataframe that contains all the variables referenced in the <code>model</code> argument, which are the data to be analyzed.
<code>starting.values</code>	Starting values for the regression coefficient parameters, as a vector. The parameter ordering matches the ordering of the formulas in the <code>model</code> argument: parameters for the terms in the first formula appear first, then come parameters for the terms in the second formula, etc. In practice it will usually be better to start by letting <code>multinomRob</code> find starting values by using the <code>multinom.t</code> option, then using the results from one run as starting values for a subsequent run done with, perhaps, a larger population of operators for <code>rgenoud</code> .
<code>equality</code>	List of equality constraints. This is a list of lists of formulas. Each formula has the same format as in the model specification, and must include only a subset of the outcomes and regressors used in the model specification formulas. All the

coefficients specified by the formulas in each list will be constrained to have the same value during estimation. For example, in the following,

```
multinomRob(model=list(y1 ~ x1, y2 ~ x2, y3 ~ 0), data=dtf,
equality=list(list(y1 ~ x1 + 0, y2 ~ x2 + 0)) );
```

the model to be estimated is

```
list(y1 ~ x1, y2 ~ x2, y3 ~ 0)
```

and the coefficients of  $x_1$  and  $x_2$  are constrained equal by

```
equality=list(list(y1 ~ x1 + 0, y2 ~ x2 + 0))
```

In the equality formulas it is necessary to say + 0 so the intercepts are not involved in the constraints. If a parameter occurs in two different lists in the equality= argument, then all the parameters in the two lists are constrained to be equal to one another. In the output this is described as consolidating the lists.

genoud.parms	List of named arguments used to control the rgenoud optimizer, which is used to compute the LQD estimator.
print.level	Specify 0 for minimal printing, 1 to print more detailed information about LQD and other intermediate computations, 2 to print details about the tanh computations, or 3 to print details about starting values computations.
iter	TRUE means to iterate between LQD and tanh estimation steps until either the algorithm converges, the number of iterations specified by the maxiter argument is reached, or if an LQD step occurs that produces a larger value than the previous step did for the overdispersion scale parameter. This option is often improves the fit of the model.
maxiter	The maximum number of iterations to be done between LQD and tanh estimation steps.
multinom.t	1 means use the multinomial multivariate-t model to compute starting values for the coefficient parameters. But if the MNL results are better (as judged by the LQD fit), MNL values will be used instead. 0 means use nonrobust maximum likelihood estimates for a multinomial regression model. 2 forces the use of the multivariate-t model for starting values even if the MNL estimates provide better starting values for the LQD. Note that with multinom.t=1 or multinom.t=2, multivariate-t starting values will not be used if the model cannot generate valid standard errors. To force the use of multivariate-t estimates even in this circumstance, see the multinom.t.df argument. If the starting.values argument is not NULL, the starting values given in that argument are used and the multinom.t argument is ignored. Multinomial multivariate-t starting values are not available when the number of outcome alternatives varies over the observations.
multinom.t.df	NA means that the degrees of freedom (DF) for the multivariate-t model (when used) should be estimated. If multinom.t.df is a number, that number will be used for the degrees of freedom and the DF will not be estimated. Only a positive number should be used. Setting multinom.t.df to a number also implies that, if multinom.t=1 or multinom.t=2, the multivariate-t starting values will be used (depending on the comparison with the MNL estimates if multinom.t=1 is set) even if the standard errors are not defined.

`MLEonly` If `TRUE`, then only the standard maximum-likelihood MNL model is estimated. No robust estimation model and no overdispersion parameter is estimated.

### Details

The tanh estimator is a redescending M-estimator, and the LQD estimator is a generalized S-estimator. The LQD is used to estimate the scale of the overdispersion. Given that scale estimate, the tanh estimator is used to estimate the coefficient parameters of the linear predictors of the multinomial regression model.

If starting values are not supplied, they are computed using a multinomial multivariate-t model. The program also computes and reports nonrobust maximum likelihood estimates for the multinomial regression model, reporting sandwich estimates for the standard errors that are adjusted for a nonrobust estimate of the error dispersion.

### Value

`multinomRob` returns a list of 15 objects. The returned objects are:

`coefficients` The tanh coefficient estimates in matrix format. The matrix has one column for each formula specified in the `model` argument. The name of each column is the name used for the count variable in the corresponding formula. The label for each row of the matrix gives the names of the regressors to which the coefficient values in the row apply. The regressor names in each label are separated by a forward slash (/), and `NA` is used to denote that no regressor is associated with the corresponding value in the matrix. The value 0 is used in the matrix to fill in for values that do not correspond to a `model` formula regressor.

`se` The tanh coefficient estimate standard errors in matrix format. The format and labelling used for the matrix is the same as is used for the `coefficients`. The standard errors are derived from the estimated asymptotic sandwich covariance estimate.

`LQDsigma2` The LQD dispersion (variance) parameter estimate. This is the LQD estimate of the scale value, squared.

`TANHsigma2` The tanh dispersion parameter estimate.

`weights` The matrix of tanh weights for the orthogonalized residuals. The matrix has one row for each observation in the data and as many columns as there are formulas specified in the `model` argument. The first column of the matrix has names for the observations, and the remaining columns contain the weights. Each of the latter columns has a name derived from the name of one of the count variables named in the `model` argument. If `count1` is the name of the count variable used in the first formula, then the second column in the matrix is named `weights:count1`, etc.

If an observation has negative values specified for some outcome variables, indicating that those outcome alternatives are not available for that observation, then values of `NA` appear in the weights matrix for that observation, as many `NA` values as there are unavailable alternatives. The `NA` values will be the last values in the affected row of the weights matrix, regardless of which outcome alternatives were unavailable for the observation.

<code>Hdiag</code>	Weights used to fully studentize the orthogonalized residuals. The matrix has one row for each observation in the data and as many columns as there are formulas specified in the <code>model</code> argument. The first column of the matrix has names for the observations, and the remaining columns contain the weights. Each of the latter columns has a name derived from the name of one of the count variables named in the <code>model</code> argument. If <code>count1</code> is the name of the count variable used in the first formula, then the second column in the matrix is named <code>Hdiag:count1</code> , etc. If an observation has negative values specified for some outcome variables, indicating that those outcome alternatives are not available for that observation, then values of 0 appear in the weights matrix for that observation, as many 0 values as there are unavailable alternatives. Values of 0 that are created for this reason will be the last values in the affected row of the weights matrix, regardless of which outcome alternatives were unavailable for the observation.
<code>prob</code>	The matrix of predicted probabilities for each category for each observation based on the tanh coefficient estimates.
<code>residuals.rotate</code>	Matrix of studentized residuals which have been made comparable by rotating each choice category to the first position. These residuals, unlike the student and standard residuals below, are no longer orthogonalized because of the rotation. These are the residuals displayed in Table 6 of the reference article.
<code>residuals.student</code>	Matrix of fully studentized orthogonalized residuals.
<code>residuals.standard</code>	Matrix of orthogonalized residuals, standardized by dividing by the overdispersion scale.
<code>mn1</code>	List of nonrobust maximum likelihood estimation results from function <a href="#">multinomMLE</a> .
<code>multinomT</code>	List of multinomial multivariate-t estimation results from function <a href="#">multinomT</a> .
<code>genoud</code>	List of LQD estimation results obtained by <code>rgenoud</code> optimization, from function <code>genoudRob</code> .
<code>mtanh</code>	List of tanh estimation results from function <code>mGNtanh</code> .
<code>error</code>	Exit error code, usually from function <code>mGNtanh</code> .
<code>iter</code>	Number of LQD-tanh iterations.

**Author(s)**

Walter R. Mebane, Jr., Cornell University, [wrm1@cornell.edu](mailto:wrm1@cornell.edu), <http://macht.arts.cornell.edu/wrm1/>

Jasjeet S. Sekhon, UC Berkeley, [sekhon@berkeley.edu](mailto:sekhon@berkeley.edu), <http://sekhon.berkeley.edu/>

**References**

Walter R. Mebane, Jr. and Jasjeet Singh Sekhon. 2004. "Robust Estimation and Outlier Detection for Overdispersed Multinomial Models of Count Data." *American Journal of Political Science* 48 (April): 391–410. <http://sekhon.berkeley.edu/multinom.pdf>

For additional documentation please visit <http://sekhon.berkeley.edu/robust/>.

**Examples**

```

# make some multinomial data
x1 <- rnorm(50);
x2 <- rnorm(50);
p1 <- exp(x1)/(1+exp(x1)+exp(x2));
p2 <- exp(x2)/(1+exp(x1)+exp(x2));
p3 <- 1 - (p1 + p2);
y <- matrix(0, 50, 3);
for (i in 1:50) {
  y[i,] <- rmultinomial(1000, c(p1[i], p2[i], p3[i]));
}

# perturb the first 5 observations
y[1:5,c(1,2,3)] <- y[1:5,c(3,1,2)];
y1 <- y[,1];
y2 <- y[,2];
y3 <- y[,3];

# put data into a dataframe
dtf <- data.frame(x1, x2, y1, y2, y3);

## Set parameters for Genoud
zz.genoud.parms <- list( pop.size           = 1000,
                        wait.generations    = 10,
                        max.generations     = 100,
                        scale.domains       = 5,
                        print.level = 0
                        )

# estimate a model, with "y3" being the reference category
# true coefficient values are: (Intercept) = 0, x = 1
# impose an equality constraint
# equality constraint: coefficients of x1 and x2 are equal
mulrobE <- multinomRob(list(y1 ~ x1, y2 ~ x2, y3 ~ 0),
                      dtf,
                      equality = list(list(y1 ~ x1 + 0, y2 ~ x2 + 0)),
                      genoud.parms = zz.genoud.parms,
                      print.level = 3, iter=FALSE);
summary(mulrobE, weights=TRUE);

#Do only MLE estimation. The following model is NOT identified if we
#try to estimate the overdispersed MNL.
dtf <- data.frame(y1=c(1,1),y2=c(2,1),y3=c(1,2),x=c(0,1))
summary(multinomRob(list(y1 ~ 0, y2 ~ x, y3 ~ x), data=dtf, MLEonly=TRUE))

```

# Index

## \*Topic **distribution**

rmultinomial, [10](#)

## \*Topic **models**

Multinomial Multivariate-T  
Regression, [1](#)  
Multinomial Regression, [3](#)  
Multinomial Regression Tanh  
Estimator, [5](#)  
Multinomial Regression Tanh  
Estimator Gauss-Newton  
Optimization, [7](#)  
Robust Multinomial  
Regression, [11](#)

## \*Topic **multivariate**

rmultinomial, [10](#)

## \*Topic **optimize**

Multinomial Regression Tanh  
Estimator Gauss-Newton  
Optimization, [7](#)

## \*Topic **regression**

Multinomial Multivariate-T  
Regression, [1](#)  
Multinomial Regression, [3](#)  
Multinomial Regression Tanh  
Estimator, [5](#)  
Multinomial Regression Tanh  
Estimator Gauss-Newton  
Optimization, [7](#)  
Robust Multinomial  
Regression, [11](#)

## \*Topic **robust**

Multinomial Regression Tanh  
Estimator, [5](#)  
Multinomial Regression Tanh  
Estimator Gauss-Newton  
Optimization, [7](#)  
Robust Multinomial  
Regression, [11](#)

match.call, [3](#)

mGNtanh (*Multinomial Regression  
Tanh Estimator  
Gauss-Newton  
Optimization*), [7](#)

Multinomial Multivariate-T  
Regression, [1](#)

Multinomial Regression, [3](#)

Multinomial Regression Tanh  
Estimator, [5](#)

Multinomial Regression Tanh  
Estimator Gauss-Newton  
Optimization, [7](#)

multinomMLE, [14](#)

multinomMLE (*Multinomial  
Regression*), [3](#)

multinomRob (*Robust Multinomial  
Regression*), [11](#)

multinomT, [15](#)

multinomT (*Multinomial  
Multivariate-T  
Regression*), [1](#)

multinomTanh (*Multinomial  
Regression Tanh  
Estimator*), [5](#)

optim, [3](#)

rmultinomial, [10](#)

Robust Multinomial Regression, [11](#)