Package 'fracreg'

August 20, 2026

Version 1.1.0

Type Package

Title Fractional Response Regressions

License GPL-3

Description Provides routines for the estimation and specification analysis of fractional response models. Includes univariate one-part, two-part, and double-inflated three-part fractional models. Further incorporates estimators for panel data settings and addresses unobserved heterogeneity and endogeneity via correlated random effects and control function approaches. Extends fractional methodology to multivariate data via fractional multinomial logit models and handles high-dimensional multicollinear data via fractional ridge regression. Calculates analytical partial effects across all model types and includes generalised goodness-of-functional-form (GGOFF) and Regression Equation Specification Error Test (RESET) hypothesis tests. Methods are described in Papke and Wooldridge (1996) <doi:10.1002/(SICI)1099-1255(199611)11:6%3C619::AID-JAE418%3E3.0.CO;2-1>, Papke and Wooldridge (2008) <doi:10.1016/j.jeconom.2008.05.009>, Buis (2008) <http://maartenbuis.nl/software/likelihoodFmlogit.pdf>, Ramalho, Ramalho and Murteira (2011) <doi:10.1111/j.1467-6419.2009.00602.x>, Fang and Ma (2013) <doi:10.1080/02664763.2012.758246>, Mullahy (2015) <doi:10.1515/jem-2012-0006>, Murteira and Ramalho (2016) <doi:10.1080/07474938.2013.806849>, and Rokem and Kay (2020) <doi:10.1093/gigascience/giaa133>.

Language en-GB

Date 2026-08-08

NeedsCompilation no

Repository CRAN

URL https://sulmanolieko.github.io/fracreg/

Depends R (>= 3.5.0)

Imports stats, maxLik, ggplot2, grid, MASS

Suggests knitr, rmarkdown, testthat (>= 3.0.0), tibble, gtsummary

VignetteBuilder knitr

RoxygenNote 7.3.3

Config/testthat/edition 3

Encoding UTF-8

Author Sulman Olieko Owili [aut, cre] (ORCID:
<https://orcid.org/0000-0001-7401-5326>)

Maintainer Sulman Olieko Owili <oliekosulman@gmail.com>

Date/Publication 2026-08-20 13:00:02 UTC

Contents

`fracreg-package` *Fractional Response Regressions*

Description

Provides comprehensive tools for the estimation and specification analysis of fractional response models. It supports univariate one-part, hurdle two-part, and double-inflated three-part models. The package also incorporates estimators for panel data settings (CRE, GMM, QML) and addresses unobserved heterogeneity and endogeneity via correlated random effects and control function approaches. It supports various link functions, calculates average and conditional partial effects analytically across all model types, and provides robust specification tests such as RESET, P test, and GGOFF tests.

Details

Package:	fracreg
Type:	Package
Version:	1.1.0
Date:	2026-08-08
License:	GPL-3

Acknowledgements

This package builds upon, consolidates, and modernises the fractional regression frameworks originally implemented in the `frm`, `frmhet`, and `frmpd` R packages developed by Joaquim J.S. Ramalho. As those original packages have been deprecated and removed from the active CRAN repository, `fracreg` serves as an actively maintained successor, ensuring these econometric tools remain available to the R community.

Furthermore, we acknowledge James Ji (@f1kidd) and A. John Woodill (@johnwoodill), the authors of the `fmlogit` R package on GitHub, whose foundational work on fractional multinomial logit models inspired the implementation of `fracregmlogit`. We also extend our gratitude to Ariel Rokem and Kendrick Kay, the authors of the `fracridge` package, whose methodological contributions to fractional ridge regression are incorporated into the `fracregridge` functionalities of this package.

Author(s)

Sulman Olieko Owili <oliekosulman@gmail.com>

References

Ramalho, J. J. S. *frm: Fractional Regression Models*. R package. Formerly available on CRAN, currently archived.

Ramalho, J. J. S. *frmhet: Fractional Regression Models under Heterogeneity*. R package. Formerly available on CRAN, currently archived.

Ramalho, J. J. S. *frmpd: Fractional Regression Models for Panel Data*. R package. Formerly available on CRAN, currently archived.

Papke, L. E. and Wooldridge, J. M. (1996), "Econometric methods for fractional response variables with an application to 401(k) plan participation rates", *Journal of Applied Econometrics*, 11(6), 619-632.

Ramalho, E.A., J.J.S. Ramalho and J.M.R. Murteira (2011), "Alternative estimating and testing empirical strategies for fractional response models", *Journal of Economic Surveys*, 25(1), 19-68.

Ramalho, E. A., Ramalho, J. J. S., and Murteira, J. M. R. (2014), "A two-part fractional response model for the spatial distribution of vineyards in Portugal", *Journal of Applied Econometrics*, 29(4), 607-630.

Fang, K., & Ma, S. (2013), "Three-part model for fractional response variables with application to Chinese household health insurance coverage", *Journal of Applied Statistics*, 40(5), 925-940.

Ji, J., and Woodill, A. J., *fmlogit: Fractional Multinomial Logit*. R package repository. <https://github.com/f1kidd/fmlogit>.

Rokem, A., and Kay, K., *fracridge: Fractional Ridge Regression*. Package repository. <https://github.com/nrdg/fracridge>.

See Also

Useful links:

- https://sulmanolieko.github.io/fracreg/

coef.fracreg *Extract Model Coefficients for fracreg*

Description

Extracts the estimated coefficients from a fitted `fracreg` model.

Usage

```
## S3 method for class 'fracreg'
coef(object, ...)
```

Arguments

`object`	A fitted model object of class `fracreg`.
`...`	Further arguments passed to or from other methods.

Value

A named vector of coefficients.

coef.fracreghet *Extract Model Coefficients for fracreghet*

Description

Extracts the estimated coefficients from a fitted `fracreghet` model.

Usage

```
## S3 method for class 'fracreghet'
coef(object, ...)
```

Arguments

`object`	A fitted model object of class `fracreghet`.
`...`	Further arguments passed to or from other methods.

Value

A named vector of coefficients.

coef.fracregmlogit *Extract Model Coefficients for fracregmlogit*

Description

Extracts the estimated coefficients from a fitted `fracregmlogit` model.

Usage

```
## S3 method for class 'fracregmlogit'
coef(object, ...)
```

Arguments

`object`	A fitted model object of class `fracregmlogit`.
`...`	Further arguments passed to or from other methods.

Value

A matrix of coefficients for each choice equation.

coef.fracregpd *Extract Model Coefficients for fracregpd*

Description

Extracts the estimated coefficients from a fitted `fracregpd` model.

Usage

```
## S3 method for class 'fracregpd'
coef(object, ...)
```

Arguments

`object`	A fitted model object of class `fracregpd`.
`...`	Further arguments passed to or from other methods.

Value

A named vector of coefficients.

coef.fracregridge *Extract Model Coefficients for fracregridge*

Description

Extracts the estimated coefficients from a fitted `fracregridge` model.

Usage

```
## S3 method for class 'fracregridge'
coef(object, ...)
```

Arguments

`object`	A fitted model object of class `fracregridge`.
`...`	Further arguments passed to or from other methods.

Value

A matrix or array of coefficients.

fitted.fracreg *Extract Fitted Values for fracreg*

Description

Extracts the fitted conditional mean values from a `fracreg` model.

Usage

```
## S3 method for class 'fracreg'
fitted(object, ...)
```

Arguments

`object`	A fitted model object of class `fracreg`.
`...`	Further arguments passed to or from other methods.

Value

A numeric vector of fitted values.

fitted.fracreghet *Extract Fitted Values for fracreghet*

Description

Extracts the fitted conditional mean values from a `fracreghet` model.

Usage

```
## S3 method for class 'fracreghet'
fitted(object, ...)
```

Arguments

`object`	A fitted model object of class `fracreghet`.
`...`	Further arguments passed to or from other methods.

Value

A numeric vector of fitted values.

fitted.fracregmlogit *Extract Fitted Values, Residuals, and Predictions*

Description

Extract Fitted Values, Residuals, and Predictions

Usage

```
## S3 method for class 'fracregmlogit'
fitted(object, ...)

## S3 method for class 'fracregmlogit'
residuals(object, ...)

## S3 method for class 'fracregmlogit'
predict(object, newdata = NULL, newbeta = NULL, ...)
```

Arguments

object	A "fracregmlogit" object.
...	Additional arguments.
newdata	A new X matrix to perform model prediction. If NULL, defaults to the original dataset. X can be a vector with length k, or a matrix with k columns, where k is the number of explanatory variables in the original model.
newbeta	A new augmented matrix of coefficients that can be used to predict outcome variables. Feeds into object$coefficient, which contains the baseline coefficient. Useful for constructing confidence intervals via simulation or bootstrapping.

Value

An object of class data.frame containing numeric values where each column corresponds to one of the choice alternatives in the response variable matrix and each row corresponds to an observation. Specifically:

- fitted: Returns the estimated fitted fractional response values (choice shares or predicted conditional probabilities).
- residuals: Returns the response residuals (the actual observed shares minus the estimated fitted shares).
- predict: Returns the predicted choice shares or conditional probabilities computed from the specified model object and newdata or newbeta.

See Also

fracregmlogit

Examples

```
data("fracreg_spending")
df <- na.omit(fracreg_spending)
X = df[,2:5]
y = df[,6:11]
results1 = fracregmlogit(y, X)

# Extract fitted values
fit = fitted(results1)

# Extract residuals
res = residuals(results1)

# Predict using the first observation from the original dataset
pred = predict(results1, newdata = X[1,])
```

fitted.fracregpd *Extract Fitted Values for fracregpd*

Description

Extracts the fitted conditional mean values from a fracregpd model.

Usage

```
## S3 method for class 'fracregpd'
fitted(object, ...)
```

Arguments

object	A fitted model object of class fracregpd.
...	Further arguments passed to or from other methods.

Value

A numeric vector of fitted values.

fitted.fracregridge *Extract Fitted Values for fracregridge*

Description

Extracts the fitted conditional mean values from a fracregridge model.

Usage

```
## S3 method for class 'fracregridge'
fitted(object, ...)
```

Arguments

object	A fitted model object of class fracregridge.
...	Further arguments passed to or from other methods.

Value

A matrix or array of fitted values.

`fracreg` *Fitting Fractional Response Regressions*

Description

`fracreg` is used to fit fractional response models, which are appropriate for responses that are proportions, percentages, or fractions restricted to the [0, 1] interval. It supports standard one-part models, two-part hurdle models for modelling boundary values at 0 or 1, and three-part models for double inflation at both 0 and 1.

Usage

```
fracreg(
  y,
  x,
  x2 = x,
  linkbin,
  linkfrac,
  type = "1P",
  inflation = 0,
  intercept = TRUE,
  table = FALSE,
  variance = TRUE,
  var.type = "default",
  var.eim = TRUE,
  var.cluster,
  dfc = FALSE,
  offset = NULL,
  or = FALSE,
  level = 0.95,
  na.action = stats::na.omit,
  ...
)
```

Arguments

`y`	a numeric vector containing the values of the response variable.
`x`	a numeric matrix, with column names, containing the values of the covariates.
`x2`	a numeric matrix, with column names, containing the values of the covariates in the fractional component of two-part models if option `type = "2P"` is defined. Defaults to `x`.
`linkbin`	a description of the link function to use in the binary component of a two-part fractional response model, or a vector of two link functions for the two binary components of a three-part model (e.g. `c("logit", "probit")`). Available options: `logit`, `probit`, `cauchit`, `loglog`, `cloglog`.

linkfrac	a description of the link function to use in standard fractional response models or in the fractional component of a two-part fractional response model. Available options: logit, probit, cauchit, loglog, cloglog.
type	a description of the model to estimate: a standard one-part model (1P, the default), a two-part model (2P), the binary component of a two-part model (2Pbin), the fractional component of a two-part model (2Pfrac), or a three-part model (3P) for double boundary inflation.
inflation	a numeric value indicating which of the extreme values of 0 (the default) or 1 is the relevant boundary value for defining two-part fractional response models.
intercept	a logical value indicating whether the model should include a constant term or not.
table	a logical value indicating whether a summary table with the regression results should be printed.
variance	a logical value indicating whether the variance of the estimated parameters should be calculated. Defaults to TRUE whenever table = TRUE.
var.type	a description of the type of variance of the estimated parameters to be calculated. Options are standard (recommended for models estimated by maximum likelihood, such as the binary component of two-part models), robust (recommended for models estimated by quasi-maximum likelihood, such as standard fractional response models or the fractional component of a two-part fractional response model), cluster (recommended in the case of panel data) and default (implements the standard or robust versions as appropriate).
var.eim	a logical value indicating whether the expected information matrix should be used in the calculation of the variance. When false, the observation information matrix will be used. Defaults to TRUE.
var.cluster	a numeric vector containing the values of the variable that specifies to which cluster each observation belongs.
dfc	a logical value indicating whether a degrees of freedom correction should be applied to the covariance matrix. Defaults to FALSE.
offset	an optional numeric vector containing an offset. It must be of the same dimension as the response variable. It specifies that the variable should be included in the model with its coefficient constrained to 1.
or	a logical value indicating whether to report odds ratios. Only valid when the link function is "logit". Defaults to FALSE.
level	a numeric value between 0 and 1 indicating the confidence level for the confidence intervals. Defaults to 0.95.
na.action	A function specifying how to handle missing values, default is stats::na.omit. If NULL, no action is taken.
...	Arguments to pass to glm.

Details

fracreg estimates one-part, two-part hurdle, and three-part double-inflated fractional response models; see Ramalho, Ramalho and Murteira (2011) and Fang and Ma (2013) for details on those models.

One-Part Fractional Response Regressions (`type = "1P"`): The standard one-part model assumes that the conditional expectation of the fractional response $y_i \in [0, 1]$ is given by:

$$E(y_i|x_i) = G(x_i\beta)$$

where $G(\cdot)$ is a known non-linear link function mapping the linear predictor to the unit interval (e.g., logit, probit). The parameters β are estimated by maximising the Bernoulli-based quasi-log-likelihood function:

$$\ln L_i(\beta) = y_i \ln[G(x_i\beta)] + (1 - y_i) \ln[1 - G(x_i\beta)]$$

This estimator requires only the correct specification of the conditional mean to yield consistent parameter estimates (Papke and Wooldridge, 1996).

Two-Part Hurdle Models (`type = "2P"`): When the data exhibits a boundary mass (e.g., at $y_i = 0$), the two-part hurdle model handles the boundary values separately from the interior fractional values. Let y_i^* be a binary indicator such that $y_i^* = 1$ if $y_i > 0$ and $y_i^* = 0$ otherwise. The probability of observing a boundary value is modelled as:

$$P(y_i = 0|x_{1i}) = 1 - F(x_{1i}\gamma_1)$$

$$P(y_i > 0|x_{1i}) = F(x_{1i}\gamma_1)$$

where $F(\cdot)$ is a binary link function. Conditional on observing an interior fractional value, the response is modelled as:

$$E(y_i|x_{2i}, y_i > 0) = G(x_{2i}\beta_2)$$

The unconditional mean of the response is therefore:

$$E(y_i|x_i) = F(x_{1i}\gamma_1) \times G(x_{2i}\beta_2)$$

Three-Part Double Inflated Models (`type = "3P"`): For data containing boundary mass at both 0 and 1, the three-part model estimates two separate binary mechanisms for each boundary and a fractional component for the interior values $(0, 1)$, extending the two-part logic to double inflation (Fang and Ma, 2013).

`fracreg` uses the standard glm command to perform the estimations. Therefore, `fracreg` is essentially a convenience command, allowing estimation of several alternative fractional response models using the same command. In addition, `fracreg` provides an R-squared measure for all models (calculated as the square of the correlation coefficient between the actual and fitted values of the dependent variable), calculates the fitted values of the dependent variable in two-part models and stores the information needed to implement some very useful commands for fractional response models: fracreg.reset (RESET test), fracreg.ptest (P test), fracreg.ggoff (GGOFF tests) and fracreg.pe (partial effects).

Value

When `type = "1P" or "2Pfrac"`, `fracreg` returns a list with the following elements:

`class`	"fracreg".
`formula`	the model formula.
`type`	the name of the estimated model.

link	the name of the specified link.
method	estimation method. Currently, "QML" (quasi-maximum likelihood) for fractional components or models and"ML" (maximum likelihood) for the binary component of two-part models.
p	a named vector of coefficients.
yhat	the fitted mean values.
xbhat	the fitted mean values of the linear predictor.
converged	logical. Was the algorithm judged to have converged?
x.names	a vector containing the names of the covariates.

If variance = TRUE or table = TRUE, the previous list also contains the following elements:

p.var	a named covariance matrix.
var.type	covariance matrix type.
var.eim	logical. Was the expected information matrix used in the computation of the covariance matrix?
dfc	logical. Was a degrees of freedom correction used for the computation of the covariance matrix?

If var.type = "cluster", the list also contains the following element:

var.cluster	the variable that specifies to which cluster each observation belongs.

When type = "2Pbin", fracreg returns a similar list with the following additional element:

LL	the value of the log-likelihood.

When type = "2P", fracreg returns the previous lists, indexed by the prefixes resBIN and resFRAC, and the following additional elements:

class	"fracreg".
type	"2P".
ybase	a numeric vector containing the values of the response variable.
x2base	a numeric matrix containing the values of the covariates.
yhat2P	the overall fitted mean values.
converged	logical. Were the algorithms judged to have converged in both parts of the model?

When type = "3P", fracreg returns the previous lists, indexed by the prefixes resBIN0, resBIN1, and resFRAC, and the following additional elements:

class	"fracreg".
type	"3P".
ybase	a numeric vector containing the values of the response variable.
x2base	a numeric matrix containing the values of the covariates.
yhat3P	the overall fitted mean values.
converged	logical. Were the algorithms judged to have converged in all parts of the model?

Odds Ratios

When `or=TRUE` and the fractional link function (`linkfrac` or `link`) is `"logit"`, the model additionally computes odds ratios for the coefficients. Odds Ratios are exponentiated coefficients. The corresponding standard errors for the odds ratios are calculated using the Delta method. The confidence intervals for the odds ratios are calculated using the adjusted standard errors and the specified `level` (defaulting to 95%). Odds ratios are particularly useful in fractional logit models as they provide a direct multiplicative interpretation of the independent variable on the odds of the fractional outcome.

Author(s)

Sulman Olieko Owili <oliekosulman@gmail.com>

References

Papke, L. E. and Wooldridge, J. M. (1996), "Econometric methods for fractional response variables with an application to 401(k) plan participation rates", *Journal of Applied Econometrics*, 11(6), 619-632.

Ramalho, E.A., J.J.S. Ramalho and J.M.R. Murteira (2011), "Alternative estimating and testing empirical strategies for fractional response models", *Journal of Economic Surveys*, 25(1), 19-68.

Fang, K., & Ma, S. (2013), "Three-part model for fractional response variables with application to Chinese household health insurance coverage", *Journal of Applied Statistics*, 40(5), 925-940.

See Also

`fracreg.reset` and `fracreg.ggoff`, for specification tests.
`fracreg.ptest`, for non-nested hypothesis tests.
`fracreg.pe`, for computing partial effects.
`fracreghet`, for fitting cross-sectional fractional response models with unobserved heterogeneity.
`fracregpd`, for fitting panel data fractional response models.

Examples

```
### Empirical 401(k) Examples
data("fracreg_k401k")
y <- fracreg_k401k$prate
X <- cbind(mrate = fracreg_k401k$mrate, age = fracreg_k401k$age,
           totemp = fracreg_k401k$totemp, sole = fracreg_k401k$sole)

# 1P Model
mod <- fracreg(y, X, type="1P", linkfrac="logit")
summary(mod)

# 1P Model reporting odds ratios and 99% confidence intervals
mod <- fracreg(y, X, type="1P", linkfrac="logit", or=TRUE, level=0.99)
summary(mod)

# 2P Model (modelling mass at 1)
mod <- fracreg(y, X, type="2P", inflation=1, linkbin="logit", linkfrac="logit")
summary(mod)
```

```
# 3P Model (inject artificial 0s for demonstration)
y_3p <- y; y_3p[1:50] <- 0
mod <- fracreg(y_3p, X, type="3P", linkbin=c("logit","logit"), linkfrac="logit")
summary(mod)

### Simulated Examples

set.seed(123)
N <- 1000
x1 <- rnorm(N)
x2 <- runif(N)

# Generating a fractional dependent variable with inflation at 0 and 1
XB <- -0.5 + 0.8 * x1 + 1.2 * x2 + rnorm(N)
y_latent <- exp(XB) / (1 + exp(XB))

y <- y_latent
# Inflate at boundaries
y[y_latent < 0.2] <- 0
y[y_latent > 0.8] <- 1

X <- cbind(x1 = x1, x2 = x2)

# fracreg estimation of a logit fractional response model
mod <- fracreg(y, X, type="1P", linkfrac="logit")
summary(mod)

# fracreg estimation of the binary logit component of the two-part fractional
# regression model with y=0 as the relevant boundary value
mod <- fracreg(y, X, type="2Pbin", inflation=0, linkbin="logit")
summary(mod)

# fracreg estimation of the fractional component of the two-part fractional
# regression model with y=0 as the relevant boundary value and using a
# probit link function
mod <- fracreg(y, X, type="2Pfrac", inflation=0, linkfrac="probit")
summary(mod)

# fracreg estimation of both components of a two-part fractional response model
# with y=0 as the relevant boundary value and using a cloglog binary link
# function and a logit fractional link function
mod <- fracreg(y, X, type="2P", inflation=0, linkbin="cloglog", linkfrac="logit")
summary(mod)

# Three-part double-inflated model (y has both 0s and 1s)
mod <- fracreg(y, X, type="3P", linkbin=c("logit","probit"), linkfrac="logit")
summary(mod)
```

`fracreg.ggoff` *GGOFF Tests for Fractional Response Regressions*

Description

`fracreg.ggoff` is used to perform Generalised Goodness-Of-Functional-Form (GGOFF) tests to check the adequacy of the functional form and link specification of fractional response models.

Usage

```
fracreg.ggoff(object, version = "LM", table = FALSE, ...)
```

Arguments

`object`	an object containing the results of an `fracreg` command.
`version`	a vector containing the test versions to use. Available options: `Wald`, `LM` (the default) and, only for the binary component of two-part models, `LR`. More than one option may be chosen.
`table`	a logical value indicating whether a summary table with the test results should be printed.
`...`	Arguments to pass to glm, which is used to estimate the model under the alternative hypothesis when `version` is a vector containing `"Wald"` or `"LR"`.

Details

`fracreg.ggoff` applies the GGOFF, GOFF1 and GOOFF2 test statistics to fractional response models estimated via `fracreg`. `fracreg.ggoff` may be used to test the link specification of: (i) one-part fractional response models; (ii) the binary component of two-part fractional response models; and (iii) the fractional component of two-part fractional response models.

GGOFF Test Framework: The Generalised Goodness-of-Functional Form (GGOFF) test evaluates the adequacy of the link function $G(\cdot)$. It is based on augmenting the baseline model with specific directions of departure. The auxiliary testing equation takes the form:

$$E(y|x) = G\left(x\beta + \gamma_1 \frac{g'(x\hat{\beta})}{g(x\hat{\beta})} + \gamma_2 x\hat{\beta}\right)$$

where $g(\cdot)$ and $g'(\cdot)$ are the first and second derivatives of $G(\cdot)$ evaluated at the linear predictor $x\hat{\beta}$. The test checks $H_0 : \gamma_1 = 0, \gamma_2 = 0$. GOFF1 and GOFF2 are variants testing individual components.

When the `Wald` version is implemented, it is taken into account the option that was chosen for computing standard errors in the model under evaluation. For the `LM` version, a robust version is computed in cases (i) and (iii) and a conventional version in case (ii). See Ramalho, Ramalho and Murteira (2014) for details on the application of the GGOFF, GOFF1 and GOOFF2 tests in the fractional response framework.

Value

`fracreg.ggoff` returns a named vector with the test results.

Author(s)

Sulman Olieko Owili <oliekosulman@gmail.com>

References

Ramalho, E.A., J.J.S. Ramalho and J.M.R. Murteira (2014), "A generalized goodness-of-functional form test for binary and fractional response models", *Manchester School*, 82(4), 488-507.

Pregibon, D. (1980), "Goodness of Link Tests for Generalized Linear Models", *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 29(1), 15-24.

See Also

fracreg, for fitting fractional response models.
fracreg.reset, for asymptotically equivalent specification tests.
fracreg.ptest, for non-nested hypothesis tests.
fracreg.pe, for computing partial effects.

Examples

```
### Empirical 401(k) Examples
data("fracreg_k401k")
y <- fracreg_k401k$prate
X <- cbind(mrate = fracreg_k401k$mrate, age = fracreg_k401k$age,
           totemp = fracreg_k401k$totemp, sole = fracreg_k401k$sole)

m <- fracreg(y, X, type="1P", linkfrac="logit")
ggoff_res <- fracreg.ggoff(m)
summary(ggoff_res)

### Simulated Examples

N <- 250
u <- rnorm(N)

X <- cbind(rnorm(N),rnorm(N))
dimnames(X)[[2]] <- c("X1","X2")

ym <- exp(X[,1]+X[,2]+u)/(1+exp(X[,1]+X[,2]+u))
y <- rbeta(N,ym*20,20*(1-ym))
y[y > 0.9] <- 1

#Testing the logit specification of a standard fractional response model
#using LM and Wald versions of the GGOFF test, based on 1 or 2 fitted powers of
#the linear predictor
mod <- fracreg(y,X,linkfrac="logit")
ggoff_res <- fracreg.ggoff(mod,c("Wald","LM"))
summary(ggoff_res)

#Testing the probit specification of the binary component of a two-part fractional
#regression model using a LR-based GGOFF test
mod <- fracreg(y,X,linkbin="probit",type="2Pbin",inf=1)
ggoff_res <- fracreg.ggoff(mod,"LR")
summary(ggoff_res)
```

fracreg.pe *Fractional Response Regressions - Partial Effects*

Description

fracreg.pe is used to compute average and/or conditional partial effects in fractional response models.

Usage

```
fracreg.pe(
  object,
  APE = TRUE,
  CPE = FALSE,
  at = NULL,
  which.x = NULL,
  variance = TRUE,
  table = FALSE
)
```

Arguments

object	an object containing the results of an fracreg command.
APE	a logical value indicating whether average partial effects are to be computed.
CPE	a logical value indicating whether conditional partial effects are to be computed.
at	a numeric vector containing the covariates' values at which the conditional partial effects are to be computed or the strings "mean" (the default) or "median", in which cases the covariates are evaluated at their mean or median values (or mode, in case of dummy variables), respectively.
which.x	a vector containing the names of the covariates to which the partial effects are to be computed.
variance	a logical value indicating whether the variance of the estimated partial effects should be calculated. Defaults to TRUE whenever table = TRUE.
table	a logical value indicating whether a summary table with the results should be printed.

Details

fracreg.pe calculates partial effects for fractional response models estimated via fracreg. fracreg.pe may be used to compute average or conditional partial effects for: (i) one-part fractional response models; (ii) the binary components of two-part and three-part fractional response models; (iii) the fractional components of two-part and three-part fractional response models; and (iv) two-part and three-part fractional response models overall.

Partial Effects for Continuous Variables: For a continuous covariate x_k, the partial effect on the conditional mean $E(y|x) = G(x\beta)$ is the first derivative with respect to x_k:

$$PE_k(x) = \frac{\partial E(y|x)}{\partial x_k} = g(x\beta)\beta_k$$

where $g(\cdot)$ is the probability density function corresponding to the link function $G(\cdot)$.

Partial Effects for Discrete Variables: For a discrete or dummy covariate x_k, the partial effect is calculated as the discrete difference in the expected value when x_k changes from 0 to 1, holding all other variables x_{-k} constant:

$$PE_k(x) = G(x_{-k}\beta_{-k} + \beta_k) - G(x_{-k}\beta_{-k})$$

Average vs. Conditional Partial Effects: - **Average Partial Effects (APE):** Evaluated for each observation i in the sample and then averaged:

$$APE_k = \frac{1}{N}\sum_{i=1}^{N} PE_k(x_i)$$

- **Conditional Partial Effects (CPE):** Evaluated at a specific vector of covariate values x^* (e.g., the sample mean or median):

$$CPE_k = PE_k(x^*)$$

For calculating standard errors, it is taken into account the option that was previously chosen for estimating the model. See Ramalho, Ramalho and Murteira (2011) and Fang and Ma (2013) for details on the computation of partial effects in the fractional response framework.

Value

`fracreg.pe` returns a list with the following element:

`PE.p` a named vector of partial effects.

If `variance = TRUE` or `table = TRUE`, the previous list also contains the following element:

`PE.sd` a named vector of standard errors of the estimated partial effects.

When both average and conditional partial effects are requested, two lists containing the previous elements are returned, indexed by the prefixes `ape` and `cpe`.

Author(s)

Sulman Olieko Owili <oliekosulman@gmail.com>

References

Ramalho, E.A., J.J.S. Ramalho and J.M.R. Murteira (2011), "Alternative estimating and testing empirical strategies for fractional response models", *Journal of Economic Surveys*, 25(1), 19-68.

Fang, K., & Ma, S. (2013), "Three-part model for fractional response variables with application to Chinese household health insurance coverage", *Journal of Applied Statistics*, 40(5), 925-940.

See Also

fracreg, for fitting fractional response models.
fracreg.reset and fracreg.ggoff, for specification tests.
fracreg.ptest, for non-nested hypothesis tests.

Examples

```
### Empirical 401(k) Examples
data("fracreg_k401k")
y <- fracreg_k401k$prate
X <- cbind(mrate = fracreg_k401k$mrate, age = fracreg_k401k$age,
           totemp = fracreg_k401k$totemp, sole = fracreg_k401k$sole)

m <- fracreg(y, X, type="1P", linkfrac="logit")
pe_res <- fracreg.pe(m)
summary(pe_res)

### Simulated Examples

N <- 250
u <- rnorm(N)

X <- cbind(rnorm(N),rnorm(N))
dimnames(X)[[2]] <- c("X1","X2")

ym <- exp(X[,1]+X[,2]+u)/(1+exp(X[,1]+X[,2]+u))
y <- rbeta(N,ym*20,20*(1-ym))
y[y > 0.9] <- 1

#Computing average partial effects for a logit fractional response model
mod <- fracreg(y,X,linkfrac="logit")
pe_res <- fracreg.pe(mod)
summary(pe_res)

#Computing average partial effects for a binary logit + fractional probit
#two-part model
mod <- fracreg(y,X,linkbin="logit",linkfrac="probit",type="2P",inf=1)
pe_res <- fracreg.pe(mod)
summary(pe_res)

#Computing conditional partial effects for X2 in the logit component
#of a two-part fractional response model, with the covariates evaluated
#at their median values
mod <- fracreg(y,X,linkfrac="logit",type="2Pfrac",inf=1)
pe_res <- fracreg.pe(mod,APE=FALSE,CPE=TRUE,at="median",which.x="X2")
summary(pe_res)

#Computing average partial effects for a three-part double-inflated model
y3p <- y
y3p[1:20] <- 0
y3p[21:40] <- 1
res3p <- fracreg(y3p,X,linkbin=c("logit","probit"),linkfrac="logit",type="3P")
```

```
pe_res <- fracreg.pe(res3p)
summary(pe_res)
```

`fracreg.ptest` *P Test for Fractional Response Regressions*

Description

`fracreg.ptest` is used to perform the P test to evaluate the specification of alternative, non-nested fractional response models by testing against each other.

Usage

```
fracreg.ptest(object1, object2, version = "Wald", table = FALSE)
```

Arguments

`object1`	an object containing the results of an `fracreg` command.
`object2`	an object containing the results of another `fracreg` command.
`version`	a vector containing the test versions to use. Available options: `Wald` (the default) and `LM`. Both options may be chosen at the same time and are computed in a robust way.
`table`	a logical value indicating whether a summary table with the test results should be printed.

Details

`fracreg.ptest` applies the P test statistic proposed by Davidson and MacKinnon (1981) to fractional response models estimated via`fracreg`. `fracreg.ptest` may be used to test against each other two alternative specifications for the link function in: (i) one-part fractional response models; (ii) the binary components of two-part and three-part fractional response models; (iii) the fractional components of two-part and three-part fractional response models; and (iv) two-part and three-part fractional response models.

P Test Framework: The P test allows the comparison of non-nested models (e.g., alternative link functions or non-nested regressors). Let model 1 specify $E_1(y|x) = G(x\beta)$ and model 2 specify $E_2(y|x) = H(z\theta)$. To test model 1 against model 2, the baseline model is augmented with the difference between the fitted values:

$$E(y|x) = G\left(x\beta + \gamma\left(\hat{y}_{M2} - \hat{y}_{M1}\right)\right)$$

where $\hat{y}_{M1} = G(x\hat{\beta})$ and $\hat{y}_{M2} = H(z\hat{\theta})$. The null hypothesis that model 1 is correct is tested via $H_0 : \gamma = 0$.

In addition, `fracreg.ptest` may be used to test one-part models against two-part or three-part models and in cases where the link functions are the same but the regressors are non-nested. See Ramalho, Ramalho and Murteira (2011) for details on the application of the P test in the fractional response framework.

Value

`fracreg.reset` returns a named vector with the test results.

Author(s)

Sulman Olieko Owili <oliekosulman@gmail.com>

References

Davidson, R. and J.G. MacKinnon (1981), "Several tests for model specification on the presence of alternative hypotheses", *Econometrica*, 49(3), 781-793.

Ramalho, E.A., J.J.S. Ramalho and J.M.R. Murteira (2011), "Alternative estimating and testing empirical strategies for fractional response models", *Journal of Economic Surveys*, 25(1), 19-68.

See Also

`fracreg`, for fitting fractional response models.
`fracreg.reset` and `fracreg.ggoff`, for specification tests.
`fracreg.pe`, for computing partial effects.

Examples

```
### Empirical 401(k) Examples
data("fracreg_k401k")
y <- fracreg_k401k$prate
X <- cbind(mrate = fracreg_k401k$mrate, age = fracreg_k401k$age,
           totemp = fracreg_k401k$totemp, sole = fracreg_k401k$sole)

m1 <- fracreg(y, X, type="1P", linkfrac="logit")
m2 <- fracreg(y, X, type="1P", linkfrac="probit")
ptest_res <- fracreg.ptest(m1, m2)
summary(ptest_res)

### Simulated Examples

N <- 250
u <- rnorm(N)

X <- cbind(rnorm(N),rnorm(N))
dimnames(X)[[2]] <- c("X1","X2")

ym <- exp(X[,1]+X[,2]+u)/(1+exp(X[,1]+X[,2]+u))
y <- rbeta(N,ym*20,20*(1-ym))
y[y > 0.9] <- 1

#Testing logit versus loglog specifications for standard fractional
#regression models using a LM version of the P test
res1 <- fracreg(y,X,linkfrac="logit")
res2 <- fracreg(y,X,linkfrac="loglog")
ptest_res <- fracreg.ptest(res1,res2,"LM")
summary(ptest_res)
```

```
#Testing a logit one-part fractional response model versus a binary logit +
#fractional probit two-part model using a Wald version of the P test
res1 <- fracreg(y,X,linkfrac="logit")
res2 <- fracreg(y,X,linkbin="logit",linkfrac="probit",type="2P",inf=1)
ptest_res <- fracreg.ptest(res1,res2,"Wald")
summary(ptest_res)
```

`fracreg.reset` *RESET Test for Fractional Response Regressions*

Description

`fracreg.reset` is used to perform the Regression Equation Specification Error Test (RESET) to check the functional form and specification of fractional response models.

Usage

```
fracreg.reset(object, lastpower.vec = 3, version = "LM", table = FALSE, ...)
```

Arguments

`object`	an object containing the results of an `fracreg` command.
`lastpower.vec`	a numeric vector containing the maximum powers of the linear predictors to be used in RESET tests.
`version`	a vector containing the test versions to use. Available options: `Wald`, `LM` (the default) and, only for the binary component of two-part models, `LR`. More than one option may be chosen.
`table`	a logical value indicating whether a summary table with the test results should be printed.
`...`	Arguments to pass to glm, which is used to estimate the model under the alternative hypothesis when `version` is a vector containing `"Wald"` or `"LR"`.

Details

`fracreg.reset` applies the RESET test statistic to fractional response models estimated via `fracreg`. `fracreg.reset` may be used to test the link specification of: (i) one-part fractional response models; (ii) the binary components of two-part and three-part fractional response models; and (iii) the fractional components of two-part and three-part fractional response models.

RESET Test Framework: The Regression Equation Specification Error Test (RESET) assesses whether the link function $G(\cdot)$ and the linear index $x\beta$ are correctly specified. It tests the null hypothesis $H_0 : \gamma = 0$ in the augmented model:

$$E(y|x) = G(x\beta + \sum_{k=2}^{P} \gamma_k (x\hat{\beta})^k)$$

where P is the maximum power of the linear predictor (specified by `lastpower.vec`) and $\hat{\beta}$ are the estimated parameters from the baseline model.

When the `Wald` version is implemented, it is taken into account the option that was chosen for computing standard errors in the model under evaluation. For the `LM` version, a robust version is computed in cases (i) and (iii) and a conventional version in case (ii). See Ramalho, Ramalho and Murteira (2011) for details on the application of the RESET test in the fractional response framework.

Value

`fracreg.reset` returns a named vector with the test results.

Author(s)

Sulman Olieko Owili <oliekosulman@gmail.com>

References

Ramalho, E.A., J.J.S. Ramalho and J.M.R. Murteira (2011), "Alternative estimating and testing empirical strategies for fractional response models", *Journal of Economic Surveys*, 25(1), 19-68.

Ramsey, J.B. (1969), "Tests for Specification Errors in Classical Linear Least-Squares Regression Analysis", *Journal of the Royal Statistical Society: Series B (Methodological)*, 31(2), 350-371.

See Also

`fracreg`, for fitting fractional response models.
`fracreg.ggoff`, for asymptotically equivalent specification tests.
`fracreg.ptest`, for non-nested hypothesis tests.
`fracreg.pe`, for computing partial effects.

Examples

```
### Empirical 401(k) Examples
data("fracreg_k401k")
y <- fracreg_k401k$prate
X <- cbind(mrate = fracreg_k401k$mrate, age = fracreg_k401k$age,
           totemp = fracreg_k401k$totemp, sole = fracreg_k401k$sole)

m <- fracreg(y, X, type="1P", linkfrac="logit")
reset_res <- fracreg.reset(m)
summary(reset_res)

### Simulated Examples

N <- 250
u <- rnorm(N)

X <- cbind(rnorm(N),rnorm(N))
dimnames(X)[[2]] <- c("X1","X2")
```

```
ym <- exp(X[,1]+X[,2]+u)/(1+exp(X[,1]+X[,2]+u))
y <- rbeta(N,ym*20,20*(1-ym))
y[y > 0.9] <- 1

#Testing the logit specification of a standard fractional response model
#using LM and Wald versions of the RESET test, based on 1 or 2 fitted powers of
#the linear predictor
mod <- fracreg(y,X,linkfrac="logit")
reset_res <- fracreg.reset(mod,2:3,c("Wald","LM"))
summary(reset_res)

#Testing the probit specification of the binary component of a two-part fractional
#regression model using LR-based RESET tests with quadratic and cubic fitted
#powers of the linear predictor
mod <- fracreg(y,X,linkbin="probit",type="2Pbin",inf=1)
reset_res <- fracreg.reset(mod,3,"LR")
summary(reset_res)
```

`fracreghet` *Fitting Fractional Response Regressions under Unobserved Heterogeneity*

Description

`fracreghet` is used to fit fractional response models under unobserved heterogeneity, i.e. regression models for proportions, percentages or fractions that suffer from neglected heterogeneity and/or endogeneity issues.

Usage

```
fracreghet(
  y,
  x,
  z = x,
  var.endog,
  start,
  type = "GMMx",
  link = "logit",
  intercept = TRUE,
  table = FALSE,
  variance = TRUE,
  var.type = "robust",
  var.cluster,
  adjust = 0,
  offset = NULL,
  or = FALSE,
  level = 0.95,
  na.action = stats::na.omit,
  ...
)
```

Arguments

y	a numeric vector containing the values of the response variable.
x	a numeric matrix, with column names, containing the values of all covariates (exogenous and endogenous).
z	a numeric matrix, with column names, containing the values of all exogenous variables (covariates and instrumental variables). Defaults to x.
var.endog	a numeric vector containing the values of the endogenous covariate (or of some transformation of it), which will be used as dependent variable in the linear reduced form assumed for application of xv-type estimators.
start	a numeric vector containing the initial values for the parameters to be optimised. Optional.
type	a description of the estimator to compute: GMMx (the default), GMMxv, GMMz, LINx, LINxv, LINz or QMLxv.
link	a description of the link function to use. Available options for all estimators: logit and cloglog. Additional available options for QML and LIN estimators: probit, cauchit and loglog.
intercept	a logical value indicating whether the model should include a constant term or not.
table	a logical value indicating whether a summary table with the regression results should be printed.
variance	a logical value indicating whether the variance of the estimated parameters should be calculated. Defaults to TRUE whenever table = TRUE.
var.type	a description of the type of variance of the estimated parameters to be calculated. Options are robust, the default, and cluster.
var.cluster	a numeric vector containing the values of the variable that specifies to which cluster each observation belongs.
adjust	the numeric value to be added to the response variable in case of boundary observations when the LIN estimators are applied or the string drop, which implies that the boundary observations are dropped.
offset	an optional numeric vector containing an offset. It must be of the same dimension as the response variable. It specifies that the variable should be included in the model with its coefficient constrained to 1.
or	a logical value indicating whether to report odds ratios. Only valid when the link function is "logit". Defaults to FALSE.
level	a numeric value between 0 and 1 indicating the confidence level for the confidence intervals. Defaults to 0.95.
na.action	A function specifying how to handle missing values, default is stats::na.omit. If NULL, no action is taken.
...	Arguments to pass to nlminb.

Details

`fracreghet` computes the GMM estimators proposed in Ramalho and Ramalho (2017) for fractional response models with unobserved heterogeneity: GMMx, which allows for neglected heterogeneity but not for endogeneity; GMMxv, which allows both issues and assumes a linear reduced form for the endogeneous covariate (or for a transformation of it); and GMMz, which also allows for both issues but does not require the assumption of a reduced form for the endogenous covariate. In addition, `fracreghet` also computes three linearised estimators (LINx, LINxv and LINz) that have similar features to their GMM counterparts. It also provides a QML estimator (QMLxv) that addresses endogeneity using a Control Function (CF) approach, which includes the first-stage reduced-form residuals as an additional regressor in the main fractional equation, providing a Hausman-type test for endogeneity.

Control Function (CF) Approach - QMLxv: When a continuous regressor y_{2i} is endogenous, the CF approach (Papke and Wooldridge, 2008; Terza et al., 2008) uses a two-stage procedure. First, a linear reduced form is estimated:

$$y_{2i} = z_i\pi + v_i$$

where z_i includes all exogenous variables and external instruments. The residuals $\hat{v}_i$ are then included in the fractional response model:

$$E(y_{1i}|z_i, y_{2i}, v_i) = G(x_i\beta + \gamma\hat{v}_i)$$

A test of $H_0 : \gamma = 0$ serves as a robust Hausman-type test for endogeneity.

Generalised Method of Moments (GMM): For estimators like GMMz, which do not strictly require a linear reduced form, the estimation relies on population orthogonality conditions between the instruments Z_i and the model residuals:

$$E[Z_i(y_i - G(x_i\beta))] = 0$$

or via specific transformations of the dependent variable to eliminate unobserved heterogeneity (Ramalho and Ramalho, 2017).

For overidentified models, `fracreghet` calculates Hansen's J statistic. For `GMMx` and `LINx`, `fracreghet` stores the information needed to implement the RESET test (fracreghet.reset). For all estimators, `fracreghet` stores the information needed to calculate partial effects (fracreghet.pe).

Value

`fracreghet` returns a list with the following elements:

`class`	"fracreghet".
`formula`	the model formula.
`type`	the name of the estimator computed.
`link`	the name of the specified link.
`adjust`	The value or the type of the adjustment applied to LIN estimators.
`p`	a named vector of coefficients.
`Hy`	the transformed values of the response variable when GMM or LIN estimators are computed or the values of the response variable in the QML case.

`xbhat`	the fitted mean values of the linear predictor (for xv-type estimators, includes the term relative to the first-stage residual).
`converged`	logical. Was the algorithm judged to have converged?
`x.names`	a vector containing the names of the covariates.

In case of an overidentifying model, the following element is also returned:

`J`	the result of Hansen's J test of overidentifying moment conditions.

If `variance = TRUE` or `table = TRUE` and the algorithm converged successfully, the previous list also contains the following elements:

`p.var`	a named covariance matrix.
`var.type`	covariance matrix type.

If `var.type = "cluster"`, the list also contains the following element:

`var.cluster`	the variable that specifies to which cluster each observation belongs.

Odds Ratios

When `or=TRUE` and the fractional link function (`linkfrac` or `link`) is `"logit"`, the model additionally computes odds ratios for the coefficients. Odds Ratios are exponentiated coefficients. The corresponding standard errors for the odds ratios are calculated using the Delta method. The confidence intervals for the odds ratios are calculated using the adjusted standard errors and the specified `level` (defaulting to 95%). Odds ratios are particularly useful in fractional logit models as they provide a direct multiplicative interpretation of the independent variable on the odds of the fractional outcome.

Author(s)

Sulman Olieko Owili <oliekosulman@gmail.com>

References

Papke, L. E. and Wooldridge, J. M. (2008), "Panel data methods for fractional response variables with an application to test pass rates", *Journal of Econometrics*, 145, 121-133.

Ramalho, E. A., & Ramalho, J. J. S. (2017), "Moment-based estimation of nonlinear regression models with boundary outcomes and endogeneity, with applications to nonnegative and fractional responses", *Econometric Reviews*, 36(4), 397-420.

Terza, J. V., Basu, A., and Rathouz, P. J. (2008), "Two-stage residual inclusion estimation: addressing endogeneity in health econometric modeling", *Journal of Health Economics*, 27(3), 531-543.

See Also

`fracreghet.reset`, for the RESET test.
`fracreghet.pe`, for computing partial effects.
`fracreg`, for fitting standard cross-sectional fractional response models.
`fracregpd`, for fitting panel data fractional response models.

Examples

```
### Empirical 401(k) Examples
data("fracreg_k401k")
y <- fracreg_k401k$prate
X_het <- cbind(mrate = fracreg_k401k$mrate, ltotemp = fracreg_k401k$ltotemp)

# fracreghet estimators do not allow exact 1s or 0s
y_adj <- y
y_adj[y_adj == 1] <- 0.999

# Instrument mrate using age

Z_emp <- cbind(age = fracreg_k401k$age, ltotemp = fracreg_k401k$ltotemp)
mod <- fracreghet(y_adj, X_het, Z_emp, var.endog = X_het[, "mrate"], type="QMLxv", link="logit")
summary(mod)

# Compute the same QMLxv estimator reporting Odds Ratios with 90% confidence intervals
mod <- fracreghet(y_adj, X_het, Z_emp, var.endog = X_het[, "mrate"], type="QMLxv",
           link="logit", or=TRUE, level=0.90)

### Simulated Examples

set.seed(123)
N <- 1000
x1 <- rnorm(N)

# Simulating an endogenous variable (var.endog) and an instrument (z1)
z1 <- rnorm(N)
u <- 0.5 * z1 + rnorm(N)
var.endog <- 0.8 * z1 + u
y_endog <- exp(0.5 * x1 + 1.2 * var.endog + u) / (1 + exp(0.5 * x1 + 1.2 * var.endog + u))

# Avoid exact 0 or 1 boundaries for some estimators
y_endog[y_endog <= 0] <- 0.01
y_endog[y_endog >= 1] <- 0.99

X <- cbind(x1 = x1, var.endog = var.endog)
Z <- cbind(x1 = x1, z1 = z1)

# Exogeneity (assuming var.endog is exogenous for comparison), GMMx estimator
mod <- fracreghet(y = y_endog, x = X, type = "GMMx", link = "logit")
summary(mod)

# Endogeneity, GMMz estimator (does not require reduced form for endog)
mod <- fracreghet(y = y_endog, x = X, z = Z, type = "GMMz", link = "logit")
summary(mod)

# Endogeneity, GMMxv estimator (assumes linear reduced form for var.endog)
mod <- fracreghet(y = y_endog, x = X, z = Z, var.endog = var.endog, type = "GMMxv", link = "logit")
summary(mod)

# Endogeneity, QMLxv control function approach
```

```
mod <- fracreghet(y = y_endog, x = X, z = Z, var.endog = var.endog, type = "QMLxv", link = "logit")
summary(mod)
```

`fracreghet.pe` *Fractional Response Regressions under Unobserved Heterogeneity - Partial Effects*

Description

`fracreghet.pe` is used to compute average and/or conditional partial effects in fractional response models under unobserved heterogeneity.

Usage

```
fracreghet.pe(
  object,
  smearing = TRUE,
  APE = TRUE,
  CPE = FALSE,
  at = NULL,
  which.x = NULL,
  table = FALSE,
  variance = TRUE
)
```

Arguments

`object`	an object containing the results of an `fracreghet` command.
`smearing`	a logical value indicating whether the smearing correction is to be applied
`APE`	a logical value indicating whether average partial effects are to be computed.
`CPE`	a logical value indicating whether conditional partial effects are to be computed.
`at`	a numeric vector containing the covariates' values at which the conditional partial effects are to be computed or the strings `"mean"` (the default) or `"median"`, in which cases the covariates are evaluated at their mean or median values (or mode, in case of dummy variables), respectively.
`which.x`	a vector containing the names of the covariates to which the partial effects are to be computed.
`table`	a logical value indicating whether a summary table with the results should be printed.
`variance`	a logical value indicating whether the variance of the estimated partial effects should be calculated. Defaults to `TRUE` whenever `table = TRUE`.

Details

`fracreghet.pe` calculates partial effects for fractional response models estimated via `fracreghet`. `fracreghet.pe` may be used to compute average or conditional partial effects. These partial effects may be conditional only on observables, using the smearing estimator, or also on unobservables, setting the error term to zero.

Partial Effects under Unobserved Heterogeneity: When unobserved heterogeneity or endogeneity is present, calculating partial effects requires dealing with the unobserved error v_i. Let the conditional mean be $E(y|x, v) = G(x\beta + \gamma v)$. - **Conditional on Observables (Smearing):** The unobserved heterogeneity is integrated out over its empirical distribution. The average partial effect for a continuous variable x_k is computed as:

$$PE_k(x) = \frac{1}{N}\sum_{i=1}^{N} g(x\beta + \gamma\hat{v}_i)\beta_k$$

- **Conditional on Unobservables (Error = 0):** The partial effect is evaluated for an individual with the mean level of unobserved heterogeneity ($v = 0$):

$$PE_k(x) = g(x\beta)\beta_k$$

For discrete variables, the partial effects are calculated as the discrete differences evaluated using either the smearing approach or setting the error term to zero.

For calculating standard errors, it is taken into account the option that was previously chosen for estimating the model. See Ramalho and Ramalho (2017) for details on the computation of partial effects for fractional response models under unobserved heterogeneity.

Value

`fracreghet.pe` returns a list with the following element:

`PE.p` a named vector of partial effects.

If `variance = TRUE` or `table = TRUE`, the previous list also contains the following element:

`PE.sd` a named vector of standard errors of the estimated partial effects.

When both average and conditional partial effects are requested, two lists containing the previous elements are returned, indexed by the prefixes `ape` and `cpe`.

Author(s)

Sulman Olieko Owili <oliekosulman@gmail.com>

References

Ramalho, E. A., & Ramalho, J. J. S. (2017), "Moment-based estimation of nonlinear regression models with boundary outcomes and endogeneity, with applications to nonnegative and fractional responses", *Econometric Reviews*, 36(4), 397-420.

See Also

fracreghet, for fitting fractional response models under unobserved heterogeneity.
fracreghet.reset, for the RESET test.

Examples

```
### Empirical 401(k) Examples
data("fracreg_k401k")
y <- fracreg_k401k$prate
X_het <- cbind(mrate = fracreg_k401k$mrate, ltotemp = fracreg_k401k$ltotemp)

# fracreghet estimators do not allow exact 1s or 0s
y_adj <- y
y_adj[y_adj == 1] <- 0.999

# Instrument mrate using age

Z_emp <- cbind(age = fracreg_k401k$age, ltotemp = fracreg_k401k$ltotemp)
res_emp <- fracreghet(y_adj, X_het, Z_emp, var.endog = X_het[, "mrate"],
                      type="QMLxv", link="logit")
pe_res <- fracreghet.pe(res_emp, which.x="mrate")
summary(pe_res)

### Simulated Examples

N <- 250
u <- rnorm(N)

X <- cbind(rnorm(N),rnorm(N))
dimnames(X)[[2]] <- c("X1","X2")

Z <- cbind(rnorm(N),rnorm(N),rnorm(N))
dimnames(Z)[[2]] <- c("Z1","Z2","Z3")

y <- exp(X[,1]+X[,2]+u)/(1+exp(X[,1]+X[,2]+u))

mod <- fracreghet(y,X,type="GMMx")

#Smearing estimator of average partial effects for variable X1
pe_res <- fracreghet.pe(mod,which.x="X1")
summary(pe_res)

#Naive estimator of conditional partial effects for all covariates,
#which are evaluated at X1=1 and X2=-1
pe_res <- fracreghet.pe(mod,smearing=FALSE,APE=FALSE,CPE=TRUE,at=c(1,-1))
summary(pe_res)
```

`fracreghet.reset` *RESET Test for Fractional Response Regressions under Neglected Heterogeneity*

Description

`fracreghet.reset` is used to test the specification of fractional response models estimated by GMMx or LINx.

Usage

```
fracreghet.reset(
  object,
  lastpower.vec = 3,
  version = "Wald",
  table = FALSE,
  ...
)
```

Arguments

`object`	an object containing the results of an `fracreghet` command.
`lastpower.vec`	a numeric vector containing the maximum powers of the linear predictors to be used in RESET tests.
`version`	a vector containing the test versions to use. Available options: `Wald` (the default) and `LM` (only available for `GMMx`).
`table`	a logical value indicating whether a summary table with the test results should be printed.
`...`	Arguments to pass to nlminb, which is used to estimate the model under the alternative hypothesis when `version` is equal to `"Wald"` and the null model was estimated by `GMMx`.

Details

`fracreghet.reset` applies the RESET test statistic to fractional response models estimated via `fracreghet` using the options `GMMx` or `LINx`. `fracreghet.reset` may be used to test simultaneously the validity of the link specification and the transformation applied to the response variable by each estimator.

RESET Test under Unobserved Heterogeneity: The test is based on augmenting the original model with powers of the linear predictor $x\hat{\beta}$. For GMMx, it tests $H_0 : \gamma = 0$ in the expanded moment conditions:

$$E\left[Z_i\left(H(y_i) - \exp\left(x_i\beta + \sum_{k=2}^{P}\gamma_k(x_i\hat{\beta})^k\right)E(e^{c_i})\right)\right] = 0$$

This simultaneously evaluates whether the mean function and the specific heterogeneity transformation $H(\cdot)$ are correctly specified.

It is taken into account the option that was chosen for computing standard errors in the model under evaluation. See Ramalho and Ramalho (2017) for details.

Value

`fracreghet.reset` returns a named vector with the test results.

Author(s)

Sulman Olieko Owili <oliekosulman@gmail.com>

References

Ramalho, E. A., & Ramalho, J. J. S. (2017), "Moment-based estimation of nonlinear regression models with boundary outcomes and endogeneity, with applications to nonnegative and fractional responses", *Econometric Reviews*, 36(4), 397-420.

Ramsey, J.B. (1969), "Tests for Specification Errors in Classical Linear Least-Squares Regression Analysis", *Journal of the Royal Statistical Society: Series B (Methodological)*, 31(2), 350-371.

See Also

`fracreghet`, for fitting fractional response models under unobserved heterogeneity.
`fracreghet.pe`, for computing partial effects.

Examples

```
### Empirical 401(k) Examples
data("fracreg_k401k")
y <- fracreg_k401k$prate
X_het <- cbind(mrate = fracreg_k401k$mrate, ltotemp = fracreg_k401k$ltotemp)

# fracreghet estimators do not allow exact 1s or 0s
y_adj <- y
y_adj[y_adj == 1] <- 0.999

# Instrument mrate using age

Z_emp <- cbind(age = fracreg_k401k$age, ltotemp = fracreg_k401k$ltotemp)
res_emp <- fracreghet(y_adj, X_het, type="GMMx", link="logit")
reset_res <- fracreghet.reset(res_emp)
summary(reset_res)

### Simulated Examples

N <- 250
u <- rnorm(N)

X <- cbind(rnorm(N),rnorm(N))
dimnames(X)[[2]] <- c("X1","X2")
```

```
Z <- cbind(rnorm(N),rnorm(N),rnorm(N))
dimnames(Z)[[2]] <- c("Z1","Z2","Z3")

y <- exp(X[,1]+X[,2]+u)/(1+exp(X[,1]+X[,2]+u))

mod <- fracreghet(y,X,type="GMMx")

#LM and Wald versions of the RESET test, based on 1 or 2 fitted powers of xb
reset_res <- fracreghet.reset(mod,2:3,c("Wald","LM"))
summary(reset_res)
```

`fracregmlogit` *Estimate Fractional Multinomial Logit Models*

Description

Used to estimate fractional multinomial logit models using quasi-maximum likelihood estimation following Papke and Wooldridge (1996).

Usage

```
fracregmlogit(
  y,
  X,
  beta0 = NULL,
  MLEmethod = "CG",
  maxit = 5e+05,
  abstol = 1e-05,
  cluster = NULL,
  reps = 1000,
  na.action = stats::na.omit,
  ...
)
```

Arguments

`y`	the dependent variable (N*J). Can be a matrix or a named data frame. The first column of the matrix is automatically treated as the baseline.
`X`	independent variable (N*K). Can be a matrix or a named data frame. If there is no intercept term in the X, then an intercept term is automatically added.
`beta0`	Initial value for beta used in optimisation. Uses a 1*K(J-1) vector. Default to a vector of zeros.
`MLEmethod`	Method of optimisation. Goes into `maxLik(method=MLEmethod)`. Choose from "NR","BFGS","CG","BHHH","SANN",or "NM". Default to "CG", the conjugate gradients method. See Details.
`maxit`	Maximum number of iterations.

abstol	Tolerance.
cluster	A vector of clusters to be used for clustered standard error computation. Default to NULL, no cluster computed.
reps	Number of bootstrap replications to be computed for clustered standard errors.
na.action	A function specifying how to handle missing values, default is stats::na.omit. If NULL, no action is taken.
...	additional parameters that go into maxLik()

Details

The fractional multinomial logit model is the expansion of the multinomial logit to fractional responses. Unlike standard multinomial logit models, which only consider 0-1 responses, the fractional multinomial logit model considers the case where the response variable is fractions that sum up to one. Examples of this type of data include percentages of budget spent in education, defence, public health; fractions of a population that have middle school, high school, college, or post-college education, etc.

This function follows Papke and Wooldridge (1996)'s paper, in which they proposed a quasi-maximum likelihood estimator for fractional response data. The likelihood function used here is a standard multinomial likelihood function, see Buis (2008) and <http://maartenbuis.nl/software/likelihoodFmlogit.pdf> for the likelihood used here. Robust standard errors are provided following Papke and Wooldridge (1996), in which they proposed an asymptotically consistent estimator of variance.

Maximisation is done by calling maxLik. maxLik is a wrapper function for different maximisation methods in R. These include most methods provided by maxLik, but also other methods such as BHHH (Berndt-Hall-Hall-Hausman).

MLE convergence can be a problem in R, especially if the dataset is large with many explanatory variables. It is recommended to call CG (Conjugate Gradients) or BHHH (Berndt-Hall-Hall-Hausman). The conjugate gradients method is usually faster, but could lead to non-convergence under certain scenarios. BHHH is slower, but has better convergence properties.

Value

The function returns an object of class "fracregmlogit". Use fracregmlogit.pe, predict, residuals, fitted to extract various useful features of the value returned by fracregmlogit.

An object of class "fracregmlogit" contains the following components:

estimates A list of matrices containing parameter estimates, standard errors, and hypothesis testing results.

baseline The baseline choice

likelihood The likelihood value

conv_code Convergence diagnostics code.

convergence Convergence messages.

count Provides dataset information

y The dependent variable data frame.

X The independent variable data frame. Augmented by factor dummy transformation , constant term added.

`rowNo` A vector of row numbers from the original X and y that is used for estimation.

`coefficient` Matrix of estimated coefficients. Augmented with the baseline coefficient (which is a vector of zeros).

`vcov` A list of matrices containing the robust variance covariance matrix for each choice variable.

`cluster` The vector of clusters.

`reps` Number of bootstrap replications for clustered standard error

References

Papke, L. E. and Wooldridge, J. M. (1996), Econometric methods for fractional response variables with an application to 401(k) plan participation rates. J. Appl. Econ., 11: 619-632.

Buis, M. L. (2008), fmlogit: Stata module fitting a fractional multinomial logit model by quasi maximum likelihood. Statistical Software Components, Boston College Department of Economics.

Mullahy, J. (2015), Multivariate fractional regression estimation of econometric share models. Journal of Econometric Methods, 4(1): 71-100.

Murteira, J. M. R., and Ramalho, J. J. S. (2016), Regression analysis of multivariate fractional data. Econometric Reviews, 35(4): 515-552.

Ji, J., and Woodill, A. J., fmlogit: Fractional Multinomial Logit. R package repository. <https://github.com/f1kidd/fmlogit>.

See Also

`fracregmlogit.pe` for computing partial effects, `plot.fracregmlogit.pe` for plotting effects, `fitted.fracregmlogit` for residuals and predictions.

Examples

```
data("fracreg_spending")
X = fracreg_spending[,2:5]
y = fracreg_spending[,6:11]

# Fit the fractional multinomial logit model
results1 = fracregmlogit(y, X)

# View estimates
summary(results1)

# Compute marginal effects
pe = fracregmlogit.pe(results1, effect="marginal", marg.type="aveacr", se=TRUE, R=50)
summary(pe)

# Plot effects for 'houseval'
plot(pe, varlist="houseval")
```

`fracregmlogit.pe` *Fractional Multinomial Logit Average Partial Effects*

Description

Calculate average partial effects (APE) of independent variables from a fractional multinomial logit model.

Usage

```
fracregmlogit.pe(
  object,
  effect = c("marginal", "discrete"),
  marg.type = "atmean",
  se = TRUE,
  varlist = NULL,
  at = NULL,
  R = 1000
)
```

Arguments

`object`	A "fracregmlogit" object.
`effect`	Can be "marginal", for marginal effects; or "discrete", for discrete changes from the min to the max.
`marg.type`	Type of marginal or discrete effects to be computed. Default to "atmean", the effect at the mean of all covariates. Also takes "aveacr", the averaged effects across all observations. See details.
`se`	Whether to calculate standard errors for those margins. Default to TRUE. See details.
`varlist`	A string vector which provides the names of variables to calculate the marginal effect for. If missing, all variables except the (Intercept) will be calculated. Use "(Intercept)" if you wish to compute the marginal effect of the intercept.
`at`	Specify values of the X-matrix at which the partial effect will be retrieved. Expects a vector input of length K-1. Only supported for `marg.type="atmean"`. See `predict.fracregmlogit(newdata)`.
`R`	Number of times to sample for the Krinsky-Robb standard error. Default to 1000.

Details

This module calculates the average partial effects (APEs) from a fractional multinomial logit model. Partial effects are the counterpart of the marginal effects in a linear model setting. In linear models, usually the parameter estimate itself represents the marginal effect (if the variable in question is continuous). In logit models, however, the parameter estimate at hand is the effect on the log-ratio

between the choice variable and the baseline variable. This function is intended to extract APEs from the coefficient estimates computed from the fractional multinomial logit models.

This function allows for two types of partial effects: marginal effects, and discrete effects. A marginal effect represents how a unit change in one continuous variable x may influence the choice variable y. The estimation of marginal effects is very straightforward. However, special care is needed when averaging the marginal effect across observations to acquire the APE. One approach is to use the estimate of the marginal effect while setting other explanatory variables at the mean. We call this the marginal effect at the mean (MEM), which corresponds to the option `marg.type="atmean"`. Another approach is to take the average of marginal effects for each individual. We call this the average marginal effect (AME), which corresponds to the option `marg.type="aveacr"`.

The discrete effect represents how a discrete change in one specific x, discrete or continuous, influences the choice variable y. This is more useful for categorical variables, as calculating the "marginal effect" makes little sense for them. In this function, we calculate the discrete effect by changing the explanatory variable from its minimum to its maximum. For a binary variable, this is just the difference between 0 and 1. Similar to the marginal effect case, we also have the discrete effect at the mean (DEM), corresponding to `marg.type="atmean"` and the average discrete effect (ADE), corresponding to `marg.type="aveacr"`.

Standard errors are provided for the effects by using the Krinsky-Robb (KR) method. Krinsky-Robb is a simulation-based method that calculates the empirical value of a function given a known distribution of its variables. Here we provide Krinsky-Robb standard errors for MEM and DEM, and the user can specify how many times of simulation `R` the Krinsky-Robb algorithm should run.

The user can also specify a subset of explanatory variables when calculating effects. This is done through specifying string vectors containing the column names of the explanatory variables to `varlist`. As the KR standard error can be computationally intensive, it is advised to calculate it only for the variables of interest.

Value

The function returns an object of class "fracregmlogit.pe". It contains the following components:

`effects` A matrix of calculated effects.

`se` A matrix of standard errors corresponding to the effects. Shows up if se=TRUE for the input parameter.

`ztable` A list of matrices containing effects, standard errors, z-stats and p-values.

`R` Number of simulation times for Krinsky-Robb standard error calculation. Null if se=FALSE.

`expl` String message explaining the effects calculated.

See Also

`fracregmlogit` for the model estimation, `plot.fracregmlogit.pe` for plotting effects.

Examples

```
data("fracreg_spending")
X = fracreg_spending[,2:5]
y = fracreg_spending[,6:11]
results1 = fracregmlogit(y, X)
```

```
# Calculate marginal effects at the mean (without standard errors for speed)
pe_marg = fracregmlogit.pe(results1, effect="marginal", se=FALSE)

# Calculate discrete effects for specific variables with standard errors
pe_disc = fracregmlogit.pe(results1, effect="discrete",
                           varlist = colnames(results1$X)[c(1,3)],
                           se=TRUE, R=50)
summary(pe_disc)
```

---

fracregpd *Fitting Panel Data Fractional Response Regressions*

Description

fracregpd is used to fit panel data regression models when the dependent variable has a bounded, fractional nature.

Usage

```
fracregpd(
  id,
  time,
  y,
  x,
  z,
  var.endog,
  x.exogenous = TRUE,
  lags,
  start,
  type,
  GMMww.cor = TRUE,
  link = "logit",
  intercept = TRUE,
  table = FALSE,
  variance = TRUE,
  var.type = "cluster",
  tdummies = FALSE,
  bootstrap = FALSE,
  B = 200,
  offset = NULL,
  or = FALSE,
  level = 0.95,
  na.action = stats::na.omit,
  ...
)
```

Arguments

id	a numeric vector identifying the cross-sectional units.
time	a numeric vector identifying the time periods in which the cross-sectional units were observed.
y	a numeric vector containing the values of the response variable.
x	a numeric matrix, with column names, containing the values of all covariates (exogenous and endogenous).
z	a numeric matrix, with column names, containing the values of all exogenous variables (covariates and external instrumental variables). Only required in case of endogenous explanatory variables.
var.endog	a numeric vector containing the values of the endogenous covariate (or of some transformation of it), which will be used as dependent variable in the linear reduced form assumed for application of the QMLcre estimator. Only required for this estimator.
x.exogenous	a logical value indicating whether all explanatory variables are assumed to be exogenous or not.
lags	a logical value indicating whether the first lags of x or z should be used as instruments for x. Defaults to TRUE for the GMMww and GMMc estimators and to FALSE for the remaining estimators. The GMMcre and QMLcre estimators do not admit lagged instruments.
start	a numeric vector containing the initial values for the parameters to be optimised. Optional.
type	a description of the estimator to compute: GMMww, GMMc, GMMbgw, GMMpfe, GMMcre, GMMpre or QMLcre.
GMMww.cor	a logical value indicating whether each explanatory variable should be transformed in deviations from its overall mean before computing the GMMww estimator.
link	a description of the link function to use. Available options for all GMM estimators: logit and cloglog. Only option for the QMLcre estimator: probit.
intercept	a logical value indicating whether the model should include a constant term or not. Only relevant for the GMMpre estimator.
table	a logical value indicating whether a summary table with the regression results should be printed.
variance	a logical value indicating whether the variance of the estimated parameters should be calculated. Defaults to TRUE whenever table = FALSERUE.
var.type	a description of the type of variance of the estimated parameters to be calculated. Options are cluster, the default, and robust. In overidentified models, it also affects the parameter estimates via the GMM weighting matrix.
tdummies	a logical value indicating whether time dummies should be included among the model explanatory variables.
bootstrap	a logical value indicating whether bootstrap should be used in the estimation of the parameter standard errors.
B	the number of bootstrap replications.

offset	an optional numeric vector containing an offset. It must be of the same dimension as the response variable. It specifies that the variable should be included in the model with its coefficient constrained to 1.
or	a logical value indicating whether to report odds ratios. Only valid when the link function is "logit". Defaults to FALSE.
level	a numeric value between 0 and 1 indicating the confidence level for the confidence intervals. Defaults to 0.95.
na.action	A function specifying how to handle missing values, default is stats::na.omit. If NULL, no action is taken.
...	Arguments to pass to nlminb.

Details

fracregpd computes the GMM estimators proposed in Ramalho, Ramalho and Coelho (2018) for panel data fractional response models with both time-variant and time-invariant unobserved heterogeneity and endogeneous covariates: GMMww, GMMc, GMMbgw, GMMpfe, GMMcre and GMMpre. In addition, fracregpd also computes QMLcre, which was proposed by Papke and Wooldridge (2008) and Wooldridge (2019).

Correlated Random Effects (CRE) - QMLcre: In panel data, unobserved individual-specific heterogeneity c_i may be correlated with the covariates x_{it}. The CRE approach (Papke and Wooldridge, 2008) models this dependence by projecting c_i onto the time averages of the strictly exogenous covariates $\bar{x}_i$:

$$c_i = \psi + \bar{x}_i\xi + a_i$$

where a_i is an error term independent of x_i. Assuming $a_i|x_i \sim N(0, \sigma_a^2)$ and a probit link, integrating out a_i yields the "population-averaged" or scaled conditional mean:

$$E(y_{it}|x_i) = G(x_{it}\beta_a + \psi_a + \bar{x}_i\xi_a)$$

where the parameters with subscript a are scaled by $(1 + \sigma_a^2)^{-1/2}$. This equation is estimated via pooled Bernoulli QML.

Generalised Method of Moments (GMM): For models where strict exogeneity fails or the link function is an exponential-type link, Ramalho et al. (2018) propose GMM estimators based on the following general moment conditions:

$$E[Z_{it}(H(y_{it}) - \exp(x_{it}\beta + c_i))] = 0$$

where $H(\cdot)$ is a transformation function and Z_{it} is a matrix of valid instruments. Estimators such as GMMww, GMMc, and GMMbgw use different transformations to eliminate the unobserved fixed effect c_i before applying GMM.

For overidentified models, fracregpd calculates Hansen's J statistic to test the validity of the overidentifying restrictions.

Value

fracregpd returns a list with the following elements:

type	the name of the estimator computed.

link	the name of the specified link.
p	a named vector of coefficients.
Hy	the transformed values of the response variable when GMM estimators are computed or the values of the response variable in the QML case.
converged	logical. Was the algorithm judged to have converged?

In case of an overidentifying model, the following element is also returned:

J	the result of Hansen's J test of overidentifying moment conditions.

If variance = TRUE or table = FALSERUE and the algorithm converged successfully, the previous list also contains the following elements:

p.var	a named covariance matrix.
var.type	covariance matrix type.

Odds Ratios

When or=TRUE and the fractional link function (linkfrac or link) is "logit", the model additionally computes odds ratios for the coefficients. Odds Ratios are exponentiated coefficients. The corresponding standard errors for the odds ratios are calculated using the Delta method. The confidence intervals for the odds ratios are calculated using the adjusted standard errors and the specified level (defaulting to 95%). Odds ratios are particularly useful in fractional logit models as they provide a direct multiplicative interpretation of the independent variable on the odds of the fractional outcome.

Author(s)

Sulman Olieko Owili <oliekosulman@gmail.com>

References

Papke, L. and Wooldridge, J.M. (2008), "Panel data methods for fractional response variables with an application to test pass rates", *Journal of Econometrics*, 145(1-2), 121-133.

Ramalho, E. A., Ramalho, J. J. S., & Coelho, L. M. S. (2018), "Exponential Regression of Fractional-Response Fixed-Effects Models with an Application to Firm Capital Structure", *Journal of Econometric Methods*, 7(1), 20150019.

Wooldridge, J. M. (2019). Correlated random effects models with unbalanced panels. *Journal of Econometrics*, 211(1), 137-150.

See Also

fracreg, for fitting standard cross-sectional fractional response models.
fracreghet, for fitting cross-sectional fractional response models with unobserved heterogeneity.

Examples

```
### Empirical 401(k) Examples
data("fracreg_k401k")
y <- fracreg_k401k$prate
X <- cbind(mrate = fracreg_k401k$mrate, age = fracreg_k401k$age,
           totemp = fracreg_k401k$totemp, sole = fracreg_k401k$sole)

# Artificial panel data structure for demonstration
N_emp <- nrow(X)
id_emp <- rep(1:(N_emp/2), each=2)
time_emp <- rep(1:2, times=N_emp/2)
mod <- fracregpd(id_emp, time_emp, y, X, type="QMLcre", link="probit")
summary(mod)

### Simulated Examples

set.seed(123)
# Simulating Panel Data
N <- 100
T_periods <- 5
id <- rep(1:N, each = T_periods)
time <- rep(1:T_periods, times = N)
x_panel <- rnorm(N * T_periods)

# Unobserved individual effect (CRE)
c_i <- rep(rnorm(N), each = T_periods)
y_panel <- exp(x_panel + c_i) / (1 + exp(x_panel + c_i))

X <- cbind(x_panel = x_panel)

# Endogenous variable and instrument simulation
z_panel <- rnorm(N * T_periods)
u_panel <- 0.5 * z_panel + rnorm(N * T_periods)
var_endog <- 0.8 * z_panel + u_panel
y_endog <- exp(x_panel + 1.2 * var_endog + c_i + u_panel) /
             (1 + exp(x_panel + 1.2 * var_endog + c_i + u_panel))

X_endog <- cbind(x_panel = x_panel, var_endog = var_endog)
Z_inst <- cbind(x_panel = x_panel, z_panel = z_panel)


# Estimate a Correlated Random Effects (CRE) Model
mod <- fracregpd(id=id, time=time, y=y_panel, x=X, type="QMLcre", link="probit")
summary(mod)

# Compute Partial Effects
pe_res <- fracregpd.pe(mod)
summary(pe_res)

# Exogeneity, no lags, no time dummies, clustered standard errors, GMMbgw estimator
mod <- fracregpd(id=id, time=time, y=y_panel, x=X, type="GMMbgw")
summary(mod)
```

```
# Estimate the GMMww estimator with odds ratios and 99% confidence intervals
mod <- fracregpd(id=id, time=time, y=y_panel, x=X, type="GMMww", or=TRUE, level=0.99)
summary(mod)

# Lagged covariates and instruments, robust standard errors, GMMww estimator
mod <- fracregpd(id=id, time=time, y=y_panel, x=X, lags=TRUE, type="GMMww", var.type="robust")
summary(mod)

# Endogeneity, time dummies, GMMpfe estimator
mod <- fracregpd(id=id, time=time, y=y_endog, x=X_endog, z=Z_inst,
                 x.exogenous=FALSE, type="GMMpfe", tdummies=TRUE)
summary(mod)
```

`fracregpd.pe` *Partial Effects for Fractional Panel Data Regression*

Description

Computes Average Partial Effects (APEs) for fractional panel data models fit using `fracregpd`. In correlated random effects (CRE) models, the time-averages of the covariates are used merely to control for unobserved heterogeneity, and as such, they are automatically filtered out when computing the structural partial effects. Standard errors are computed using the Delta method and the bootstrapped parameter variance matrix.

Usage

```
fracregpd.pe(
  object,
  APE = TRUE,
  CPE = FALSE,
  at = NULL,
  which.x = NULL,
  variance = TRUE,
  table = FALSE,
  ...
)
```

Arguments

`object`	An object of class `fracregpd`.
`APE`	logical. Compute Average Partial Effects?
`CPE`	logical. Compute Conditional Partial Effects? (Not currently supported for panel data models).
`at`	numeric vector. The values at which to evaluate the CPE.

which.x	character vector. Variables for which to compute partial effects. By default, auxiliary CRE parameters (like _mean and vhat) are automatically excluded so that partial effects are only computed for the main structural parameters.
variance	logical. Compute standard errors using the Delta method?
table	logical. Print the resulting partial effects table?
...	further arguments passed to or from other methods.

Value

An object of class fracreg.pe containing the standard coefficient tables with Average Partial Effects.

Author(s)

Sulman Olieko Owili <oliekosulman@gmail.com>

See Also

fracregpd, fracreg.pe

Examples

```
# Simulate Panel Data
N <- 50
T <- 5
id <- rep(1:N, each=T)
time <- rep(1:T, N)
x1 <- rnorm(N*T)
x2 <- rnorm(N*T)
z1 <- rnorm(N*T)
u <- rnorm(N*T)
y <- exp(x1 - x2 + u)/(1 + exp(x1 - x2 + u))
X <- cbind(x1 = x1, x2 = x2)
Z <- cbind(x1 = x1, z1 = z1)

# Estimate panel data model
mod_pd <- fracregpd(id=id, time=time, y=y, x=X, z=Z,
                    x.exogenous=FALSE, type="GMMbgw", link="logit")

# Compute Average Partial Effects
pe_res <- fracregpd.pe(mod_pd)
summary(pe_res)
```

`fracregridge` *Fractional Ridge Regression*

Description

`fracregridge` implements Fractional Ridge Regression (Rokem & Kay, 2020), which is an approach to regularized linear regression. Unlike standard ridge regression where the penalty term α is chosen directly, `fracregridge` allows you to specify the desired *fraction* of the unregularized OLS coefficient vector length. The algorithm then automatically determines the corresponding α penalties.

Usage

```
fracregridge(
  y,
  x,
  fracs = seq(0.1, 1, by = 0.1),
  tol = 1e-10,
  intercept = TRUE,
  na.action = stats::na.omit,
  ...
)
```

Arguments

`y`	A numeric vector or matrix of the dependent variable(s).
`x`	A numeric matrix of the explanatory variables.
`fracs`	A numeric vector indicating the desired fractions of the unregularized coefficient vector length. Default is `seq(0.1, 1.0, by=0.1)`. Values must be sorted in ascending order.
`tol`	A numeric tolerance under which singular values of the `x` matrix are considered to be zero. Default is `1e-10`.
`intercept`	logical. If `TRUE`, an intercept is included in the model. Default is `TRUE`.
`na.action`	A function specifying how to handle missing values, default is `stats::na.omit`. If `NULL`, no action is taken.
`...`	further arguments passed to or from other methods.

Details

Standard ridge regression minimizes the following objective function:

$$L = (y - X\beta)'(y - X\beta) + \alpha\beta'\beta$$

The penalty α shrinks the coefficients towards zero, reducing the length of the coefficient vector $||\beta||_2$. However, choosing α can be unintuitive. `fracregridge` re-parameterizes the problem so the user specifies `fracs`, the fraction of the unregularized Ordinary Least Squares (OLS) vector length $\gamma = \frac{||\beta(\alpha)||_2}{||\beta(0)||_2}$. The function automatically determines the α values corresponding to these fractions.

Value

An object of class `fracregridge` containing:

`coef`	The estimated ridge coefficients for each requested fraction.
`alphas`	The corresponding α penalty values.
`fracs`	The grid of fractions.
`call`	The matched call.

Author(s)

Sulman Olieko Owili <oliekosulman@gmail.com>

References

Rokem, A., & Kay, K. (2020). Fractional ridge regression: a fast, interpretable reparameterization of ridge regression. *GigaScience*, 9(12).

Rokem, A., and Kay, K., fracridge: Fractional Ridge Regression. Package repository. <https://github.com/nrdg/fracridge>.

See Also

`fracreg`

Examples

```
# Empirical 401(k) Example
data("fracreg_k401k")
y_401k <- fracreg_k401k$prate
X_401k <- cbind(mrate = fracreg_k401k$mrate, age = fracreg_k401k$age,
                totemp = fracreg_k401k$totemp, sole = fracreg_k401k$sole)

# Fit fractional ridge regression for the 401(k) participation rates
mod_401k <- fracregridge(y = y_401k, x = X_401k, fracs = seq(0.2, 1.0, by = 0.2))

# View full detailed summary
summary(mod_401k)

# Compute Average Partial Effects for Ridge
pe_401k <- fracregridge.pe(mod_401k)
summary(pe_401k)

# Simulated Data Example
set.seed(123)
n <- 100
p <- 10
y <- rnorm(n)
X <- matrix(rnorm(n * p), n, p)
colnames(X) <- paste0("X", 1:p)

# Fit Fractional Ridge Regression
# We want the coefficients that correspond to 30\%, 50\%, and 80\% of the OLS length
```

```
mod_sim <- fracregridge(y, X, fracs = c(0.3, 0.5, 0.8))

# View brief summary
print(mod_sim)

# Compute Partial Effects
pe_sim <- fracregridge.pe(mod_sim)
summary(pe_sim)
set.seed(123)
y <- rnorm(100)
x <- matrix(rnorm(1000), 100, 10)
colnames(x) <- paste0("x", 1:10)

# Fit fractional ridge regression
mod <- fracregridge(y, x, fracs = c(0.3, 0.5, 0.8))
print(mod)
summary(mod)
```

`fracregridge.pe` *Partial Effects for Fractional Ridge Regression*

Description

Because Fractional Ridge Regression fits a linear model without a link function, the partial effects are mathematically identical to the estimated ridge coefficients. This function serves as a wrapper to maintain API compatibility with the rest of the `fracreg` package, printing a brief notification and returning the standard coefficient tables.

Usage

```
fracregridge.pe(
  object,
  APE = TRUE,
  CPE = FALSE,
  at = NULL,
  variance = TRUE,
  table = FALSE,
  ...
)
```

Arguments

`object`	An object of class `fracregridge`.
`APE`	logical. Ignored for ridge regression.
`CPE`	logical. Ignored for ridge regression.
`at`	numeric vector. Ignored for ridge regression.
`variance`	logical. Ignored for ridge regression.
`table`	logical. Ignored for ridge regression.
`...`	further arguments passed to or from other methods.

Value

An object of class `fracreg.pe` containing the standard coefficient tables.

Author(s)

Sulman Olieko Owili <oliekosulman@gmail.com>

See Also

`fracregridge`, `fracreg.pe`

Examples

```
# Generate random data
set.seed(123)
y <- rnorm(100)
X <- matrix(rnorm(1000), 100, 10)
colnames(X) <- paste0("X", 1:10)

# Fit Fractional Ridge Regression
mod <- fracregridge(y, X, fracs = c(0.3, 0.5))

# Compute Partial Effects (identical to coefficients)
pe <- fracregridge.pe(mod)
print(pe)
```

---

fracreg_clean_data *Clean Data for Fractional Regression Models*

Description

An internal helper to gracefully drop missing values across an arbitrary number of vectors and matrices, replicating the functionality of na.action = na.omit for multi-array model inputs.

Usage

```
fracreg_clean_data(..., na.action = stats::na.omit)
```

Arguments

`...`	A variable number of vectors, matrices, or data frames.
`na.action`	A function specifying how to handle missing values, default is `stats::na.omit`. If `NULL`, no action is taken.

Value

A named list containing the subsets of the provided arrays without missing values.

`fracreg_k401k` *401(k) Plan Participation Data*

Description

A cross-sectional dataset on 401(k) plan participation rates and firm characteristics, widely used in empirical applications of fractional response models (e.g., Papke & Wooldridge, 1996). The data is derived from the 'wooldridge' package.

Usage

```
data("fracreg_k401k")
```

Format

A data frame with 1,534 observations and 10 variables:

prate Participation rate: proportion of eligible employees participating in the 401(k) plan (0 to 1).

mrate Match rate: the firm's contribution matching rate per dollar.

totpart Total number of participants.

totelg Total number of eligible employees.

age Age of the 401(k) plan in years.

totemp Total number of firm employees.

sole Indicator variable: 1 if the 401(k) is the sole retirement plan offered.

ltotemp Natural log of total employees.

age_sq Square of plan age.

mrate_sq Square of match rate.

Source

Papke, L. E., & Wooldridge, J. M. (1996). "Econometric Methods for Fractional Response Variables with an Application to 401(k) Plan Participation Rates." Journal of Applied Econometrics, 11(6), 619-632.

Examples

```
data("fracreg_k401k")
summary(fracreg_k401k$prate)
```

`fracreg_spending` *Government Spending Data*

Description

Spending on different categories by Dutch cities in 2005. This dataset is commonly used to demonstrate fractional multinomial logit models.

Usage

```
data("fracreg_spending")
```

Format

A data frame with 429 observations and 12 variables:

muni Name of municipality

houseval Average value of a house in 100,000 euros

popdens Population density in 1000s of persons per square km

noleft No left party in city government

minorityleft Minority left party in city government

governing Fraction of spending on governing

safety Fraction of spending on safety

education Fraction of spending on education

recreation Fraction of spending on recreation

social Fraction of spending on social services

urbanplanning Fraction of spending on urban planning

tot Total spending or population

Source

<http://fmwww.bc.edu/repec/bocode/c/citybudget.dta>

Examples

```
data("fracreg_spending")
head(fracreg_spending)
```

logLik.fracreg *Extract Log-Likelihood for fracreg*

Description

Extracts the log-pseudolikelihood or log-likelihood from a fitted `fracreg` model.

Usage

```
## S3 method for class 'fracreg'
logLik(object, ...)
```

Arguments

`object`	A fitted model object of class `fracreg`.
`...`	Further arguments passed to or from other methods.

Value

An object of class `logLik`.

logLik.fracreghet *Extract Log-Likelihood for fracreghet*

Description

Extracts the log-pseudolikelihood or log-likelihood from a fitted `fracreghet` model.

Usage

```
## S3 method for class 'fracreghet'
logLik(object, ...)
```

Arguments

`object`	A fitted model object of class `fracreghet`.
`...`	Further arguments passed to or from other methods.

Value

An object of class `logLik`.

logLik.fracregmlogit *Extract Log-Likelihood for fracregmlogit*

Description

Extracts the log-pseudolikelihood from a fitted `fracregmlogit` model.

Usage

```
## S3 method for class 'fracregmlogit'
logLik(object, ...)
```

Arguments

`object`	A fitted model object of class `fracregmlogit`.
`...`	Further arguments passed to or from other methods.

Value

An object of class `logLik`.

logLik.fracregpd *Extract Log-Likelihood for fracregpd*

Description

Extracts the log-pseudolikelihood or log-likelihood from a fitted `fracregpd` model.

Usage

```
## S3 method for class 'fracregpd'
logLik(object, ...)
```

Arguments

`object`	A fitted model object of class `fracregpd`.
`...`	Further arguments passed to or from other methods.

Value

An object of class `logLik`.

logLik.fracregridge *Extract Log-Likelihood for fracregridge*

Description

Extracts the log-likelihood from a fitted `fracregridge` model.

Usage

```
## S3 method for class 'fracregridge'
logLik(object, ...)
```

Arguments

`object`	A fitted model object of class `fracregridge`.
`...`	Further arguments passed to or from other methods.

Value

An object of class `logLik`.

nobs.fracreg *Extract the Number of Observations for fracreg*

Description

Extracts the number of observations used to estimate a `fracreg` model.

Usage

```
## S3 method for class 'fracreg'
nobs(object, ...)
```

Arguments

`object`	A fitted model object of class `fracreg`.
`...`	Further arguments passed to or from other methods.

Value

An integer denoting the number of observations.

nobs.fracreghet *Extract the Number of Observations for fracreghet*

Description

Extracts the number of observations used to estimate a `fracreghet` model.

Usage

```
## S3 method for class 'fracreghet'
nobs(object, ...)
```

Arguments

`object`	A fitted model object of class `fracreghet`.
`...`	Further arguments passed to or from other methods.

Value

An integer denoting the number of observations.

nobs.fracregmlogit *Extract the Number of Observations for fracregmlogit*

Description

Extracts the number of observations used to estimate a `fracregmlogit` model.

Usage

```
## S3 method for class 'fracregmlogit'
nobs(object, ...)
```

Arguments

`object`	A fitted model object of class `fracregmlogit`.
`...`	Further arguments passed to or from other methods.

Value

An integer denoting the number of observations.

nobs.fracregpd *Extract the Number of Observations for fracregpd*

Description

Extracts the number of observations used to estimate a fracregpd model.

Usage

```
## S3 method for class 'fracregpd'
nobs(object, ...)
```

Arguments

object	A fitted model object of class fracregpd.
...	Further arguments passed to or from other methods.

Value

An integer denoting the number of observations.

nobs.fracregridge *Extract the Number of Observations for fracregridge*

Description

Extracts the number of observations used to estimate a fracregridge model.

Usage

```
## S3 method for class 'fracregridge'
nobs(object, ...)
```

Arguments

object	A fitted model object of class fracregridge.
...	Further arguments passed to or from other methods.

Value

An integer denoting the number of observations.

`plot.fracregmlogit` *Plot Marginal or Discrete Effects of Willingness to Pay*

Description

Plot marginal or discrete effects of willingness to pay, potentially against another variable.

Usage

```
## S3 method for class 'fracregmlogit'
plot(
  x,
  wtp.vec = NULL,
  varlist = NULL,
  against = NULL,
  mfrow = NULL,
  t = 500,
  effect = c("discrete", "marginal"),
  type = NULL,
  plot.show = TRUE,
  ...
)
```

Arguments

`x`	A "fracregmlogit" object.
`wtp.vec`	A numeric vector for willingness to pay.
`varlist`	A string vector which provides the names of variables to plot the effect for. If missing, all variables in the object will be plotted.
`against`	A vector with the same length as the number of observations in the model, or the name of a variable. Serves as the x-axis in the plots.
`mfrow`	A numeric vector with two elements. Specifies the number of rows and columns in a panel. Similar to par(mfrow=c()). Default to NULL, and the program will choose a square panel.
`t`	Number of points to be used for smoothing.
`effect`	The type of effect ("marginal" or "discrete").
`type`	Plot type.
`plot.show`	If TRUE, the plot will be created. Otherwise, the function returns raw data that can be used to create user-specified (custom) plots.
`...`	Additional arguments.

Details

This function provides a visualisation tool for potentially heterogeneous marginal and discrete effects of willingness to pay. The function allows the user to plot marginal effects to detect any patterns in the effects, in itself and against other variables. The plot also allows visualisation of sub-groups in data, which can be very useful to visualise categorical and dummy variables.

The function takes a `fracregmlogit` object, and internally calls `wtp` and `fracregmlogit.pe` to compute the willingness to pay at different data points.

Additional parameters include `varlist`, a vector of string variable names to be plotted.

`against` allows a different variable to be chosen as the x-axis. `against` can supply the column name of a variable in the original dataset to plot against.

Value

Panel plots of effects vs. chosen variables.

See Also

`wtp`, `fracregmlogit.pe`

Examples

```
data("fracreg_spending")
X = fracreg_spending[,2:5]
y = fracreg_spending[,6:11]
results1 = fracregmlogit(y, X)

# Define a willingness to pay vector
wtp.vec = c(1, 1, 1, 1, 1, 1)

# Plot WTP for 'popdens'
plot(results1, wtp.vec=wtp.vec, varlist="popdens")
```

---

plot.fracregmlogit.pe *Plot Marginal or Discrete Effects*

Description

Plot the desired effect at each observed value for each choice.

Usage

```
## S3 method for class 'fracregmlogit.pe'
plot(
  x,
  varlist = NULL,
  X = NULL,
  y = NULL,
```

```
  against = NULL,
  against.x = NULL,
  against.y = NULL,
  group.x = NULL,
  group.algebra = NULL,
  mfrow = NULL,
  ...
)
```

Arguments

x	A "fracregmlogit.pe" object.
varlist	A string vector which provides the names of variables to plot the effect for. If missing, all variables in the object will be plotted.
X	A matrix of independent variables.
y	A matrix of dependent variables.
against	A vector with the same length as the number of observations in the model. Serves as the x-axis in the plots.
against.x	A character string, supply the column name in the X matrix to plot against.
against.y	A character string, supply the column name in the y matrix to plot against.
group.x	A character string. Supply the column name in the X matrix to group upon.
group.algebra	A character string. Supply additional algebra imposed on the group variable.
mfrow	A numeric vector with two elements. Specifies the number of rows and columns in a panel. Similar to par(mfrow=c()). Default to NULL, and the program will choose a square panel.
...	Additional arguments.

Details

This function provides a visualisation tool for potentially heterogeneous marginal and discrete effects. The function allows the user to plot marginal effects to detect any patterns in the effects, in itself and against other variables. The plot also allows visualisation of sub-groups in data, which can be very useful to visualise categorical and dummy variables.

The function takes a `fracregmlogit.pe` object, created by the `fracregmlogit.pe()` function. Note that since the plotting requires marginal effects for all observations, the object should be created by choosing `marg.type="aveacr"`, the average across method for effects calculation.

Additional parameters include `varlist`, a vector of string variable names to be plotted. `X` and `y` are the dependent and independent variable matrices in the original regression model.

`against`, `against.x`, and `against.y` allow different variables to be chosen as the x-axis. `against` directly supplies the vector to be plotted against, whereas `against.x` and `against.y` supply variable names in the original dataset. Note that the user has to provide `X` and `y` in order to use the column name options, respectively.

`group.x` supplies the column name in the X matrix to group by. The plot will be able to differentiate different groups by colours. Additionally, the user can supply a string to `group.algebra`, which provides an algebra operation that will be evaluated on the group vector. For example, choosing

group.x = "a" and group.algebra = ">0" will create two groups, one with X$a > 0, and one with X$a <= 0.

Value

Panel plots of effects vs. chosen variables.

See Also

fracregmlogit.pe

Examples

```
data("fracreg_spending")
X = fracreg_spending[,2:5]
y = fracreg_spending[,6:11]
results1 = fracregmlogit(y, X)

# Calculate marginal effects with marg.type="aveacr" (no standard errors for speed)
effect1 = fracregmlogit.pe(results1, effect="marginal", marg.type="aveacr", se=FALSE)

# Plot effects
plot(effect1, X=results1$X, against.x = "houseval", group.x = "popdens", group.algebra = ">10")
```

---

predict.fracreg *Predict Method for fracreg*

Description

Predicts conditional mean values from a fitted fracreg model.

Usage

```
## S3 method for class 'fracreg'
predict(object, newdata = NULL, ...)
```

Arguments

object	A fitted model object of class fracreg.
newdata	An optional data frame or matrix in which to look for variables with which to predict. If omitted, the fitted values are used.
...	Further arguments passed to or from other methods.

Value

A numeric vector of predicted values.

predict.fracreghet *Predict Method for fracreghet*

Description

Predicts conditional mean values from a fitted `fracreghet` model.

Usage

```
## S3 method for class 'fracreghet'
predict(object, newdata = NULL, ...)
```

Arguments

`object`	A fitted model object of class `fracreghet`.
`newdata`	An optional data frame or matrix in which to look for variables with which to predict. If omitted, the fitted values are used.
`...`	Further arguments passed to or from other methods.

Value

A numeric vector of predicted values.

predict.fracregpd *Predict Method for fracregpd*

Description

Predicts conditional mean values from a fitted `fracregpd` model.

Usage

```
## S3 method for class 'fracregpd'
predict(object, newdata = NULL, ...)
```

Arguments

`object`	A fitted model object of class `fracregpd`.
`newdata`	An optional data frame or matrix in which to look for variables with which to predict. If omitted, the fitted values are used.
`...`	Further arguments passed to or from other methods.

Value

A numeric vector of predicted values.

predict.fracregridge *Predict Method for fracregridge*

Description

Predicts conditional mean values from a fitted `fracregridge` model.

Usage

```
## S3 method for class 'fracregridge'
predict(object, newdata = NULL, ...)
```

Arguments

`object`	A fitted model object of class `fracregridge`.
`newdata`	An optional data frame or matrix in which to look for variables with which to predict. If omitted, the fitted values are used.
`...`	Further arguments passed to or from other methods.

Value

A matrix or array of predicted values.

residuals.fracreg *Extract Model Residuals for fracreg*

Description

Extracts the response residuals from a fitted `fracreg` model.

Usage

```
## S3 method for class 'fracreg'
residuals(object, ...)
```

Arguments

`object`	A fitted model object of class `fracreg`.
`...`	Further arguments passed to or from other methods.

Value

A numeric vector of residuals.

residuals.fracreghet *Extract Model Residuals for fracreghet*

Description

Extracts the response residuals from a fitted `fracreghet` model.

Usage

```
## S3 method for class 'fracreghet'
residuals(object, ...)
```

Arguments

`object`	A fitted model object of class `fracreghet`.
`...`	Further arguments passed to or from other methods.

Value

A numeric vector of residuals.

residuals.fracregpd *Extract Model Residuals for fracregpd*

Description

Extracts the response residuals from a fitted `fracregpd` model.

Usage

```
## S3 method for class 'fracregpd'
residuals(object, ...)
```

Arguments

`object`	A fitted model object of class `fracregpd`.
`...`	Further arguments passed to or from other methods.

Value

A numeric vector of residuals.

residuals.fracregridge *Extract Model Residuals for fracregridge*

Description

Extracts the response residuals from a fitted `fracregridge` model.

Usage

```
## S3 method for class 'fracregridge'
residuals(object, ...)
```

Arguments

`object`	A fitted model object of class `fracregridge`.
`...`	Further arguments passed to or from other methods.

Value

A matrix or array of residuals.

summary.fracregmlogit *Generate Summary Tables for fracregmlogit Objects*

Description

Generate tables of coefficient estimates, partial effects, and willingness to pay from fracregmlogit-type objects.

Usage

```
## S3 method for class 'fracregmlogit'
print(x, ...)

## S3 method for class 'fracregmlogit'
summary(object, ...)
```

Arguments

`x`	an object of class "fracregmlogit", "fracregmlogit.pe", or "fracregmlogit.wtp" (used by print methods).
`...`	Additional arguments passed to the printCoefmat function.
`object`	an object with class "fracregmlogit", "fracregmlogit.pe", or "fracregmlogit.wtp".

Details

This module provides summary methods for three fracregmlogit objects: `fracregmlogit`, `fracregmlogit.pe`, and `fracregmlogit.wtp`.

For `fracregmlogit` objects, the summary prints the number of observations, log pseudo-likelihood, baseline choice, and the coefficient estimates with standard errors, z-statistics, and p-values for each choice equation.

For `fracregmlogit.pe` objects, it displays the marginal or discrete effects along with their computed standard errors (if Krinsky-Robb sampling was performed) for each choice.

For `fracregmlogit.wtp` objects, it provides a table of the aggregated willingness to pay along with its standard errors and test statistics.

Value

Returns the object invisibly.

See Also

`fracregmlogit`, `fracregmlogit.pe`

Examples

```
data("fracreg_spending")
X = fracreg_spending[,2:5]
y = fracreg_spending[,6:11]

# generate fracregmlogit summary
results1 = fracregmlogit(y, X)
summary(results1)

# generate marginal effects summary
effects1 = fracregmlogit.pe(results1, effect="marginal", se=FALSE)
summary(effects1)
```

---

vcov.fracreg *Extract Covariance Matrix for fracreg*

Description

Extracts the estimated variance-covariance matrix of the parameters.

Usage

```
## S3 method for class 'fracreg'
vcov(object, ...)
```

Arguments

`object`	A fitted model object of class `fracreg`.
`...`	Further arguments passed to or from other methods.

Value

A matrix of the estimated covariances.

`vcov.fracreghet` *Extract Covariance Matrix for fracreghet*

Description

Extracts the estimated variance-covariance matrix of the parameters.

Usage

```
## S3 method for class 'fracreghet'
vcov(object, ...)
```

Arguments

`object`	A fitted model object of class `fracreghet`.
`...`	Further arguments passed to or from other methods.

Value

A matrix of the estimated covariances.

`vcov.fracregmlogit` *Extract Covariance Matrix for fracregmlogit*

Description

Extracts the estimated variance-covariance matrices of the parameters.

Usage

```
## S3 method for class 'fracregmlogit'
vcov(object, ...)
```

Arguments

`object`	A fitted model object of class `fracregmlogit`.
`...`	Further arguments passed to or from other methods.

Value

A list of covariance matrices for each choice equation.

---

vcov.fracregpd *Extract Covariance Matrix for fracregpd*

Description

Extracts the estimated variance-covariance matrix of the parameters.

Usage

```
## S3 method for class 'fracregpd'
vcov(object, ...)
```

Arguments

object	A fitted model object of class fracregpd.
...	Further arguments passed to or from other methods.

Value

A matrix of the estimated covariances.

---

vcov.fracregridge *Extract Covariance Matrix for fracregridge*

Description

Extracts the estimated variance-covariance matrices of the parameters.

Usage

```
## S3 method for class 'fracregridge'
vcov(object, ...)
```

Arguments

object	A fitted model object of class fracregridge.
...	Further arguments passed to or from other methods.

Value

A matrix of the estimated covariances.

wtp *"Willingness to Pay" for fracregmlogit models*

Description

Evaluate the "Willingness to Pay" given a set of arbitrary values for outcome variables. Usually used for policy evaluations where the total magnitude of marginal change matters.

Usage

```
wtp(object, wtp.vec, varlist = NULL, indv.obs = FALSE)
```

Arguments

object	A fracregmlogit.pe object.
wtp.vec	A numeric vector containing the arbitrary outcome values to be evaluated for each choice j.
varlist	A string vector which provides the names of variables to calculate the wtp for. If missing, all variables in the object will be calculated.
indv.obs	A logical value indicating whether to return individual observations.

Details

This function calculates the aggregate effect of a variable on the "willingness to pay" by linearly multiplying the average partial effect with ex-ante (arbitrary) willingness to pay numbers associated with each choice.

Suppose there are three choices A, B, C, each with a willingness to pay (or cost, profit, budget), of 100, 200, and 300. The discrete effects of variable X on A, B and C are 0.5, 0.5, and -1, with standard errors 0.2, 0.3 and 0.5. The aggregated discrete effect of X on the total willingness to pay (or cost), is thus 100*0.5 + 200*0.5 + 300*(-1) = -150. The standard error can also be calculated to be 162.8, assuming that the standard error is independent. A simple z-test is provided to test whether the aggregate effect is different from zero.

Note that if the input fracregmlogit.pe object has no standard error computation, then no standard error will be computed for the willingness to pay.

Value

A "fracregmlogit.wtp" object containing the estimates, standard error, z-stats, and p-value.

See Also

fracregmlogit.pe, plot.fracregmlogit

Examples

```
data("fracreg_spending")
X = fracreg_spending[,2:5]
y = fracreg_spending[,6:11]
results1 = fracregmlogit(y, X)
pe = fracregmlogit.pe(results1)
# Assuming arbitrary WTP values for the 6 choices
wtp_est = wtp(pe, wtp.vec = c(1, 2, 3, 4, 5, 6), varlist = "houseval")
summary(wtp_est)
```

Index