

# Package ‘alpaca’

October 12, 2022

**Type** Package

**Title** Fit GLM's with High-Dimensional k-Way Fixed Effects

**Version** 0.3.4

## Description

Provides a routine to partial out factors with many levels during the optimization of the log-likelihood function of the corresponding generalized linear model (glm). The package is based on the algorithm described in Stammann (2018) <[arXiv:1707.01815](https://arxiv.org/abs/1707.01815)> and is restricted to glm's that are based on maximum likelihood estimation and nonlinear. It also offers an efficient algorithm to recover estimates of the fixed effects in a post-estimation routine and includes robust and multi-way clustered standard errors. Further the package provides analytical bias corrections for binary choice models derived by Fernandez-Val and Weidner (2016) <[doi:10.1016/j.jeconom.2015.12.014](https://doi.org/10.1016/j.jeconom.2015.12.014)> and Hinz, Stammann, and Waner (2020) <[arXiv:2004.12655](https://arxiv.org/abs/2004.12655)>.

**License** GPL-3

**Depends** R (>= 3.1.0)

**Imports** data.table, Formula, MASS, Rcpp, stats, utils

**LinkingTo** Rcpp, RcppArmadillo

**URL** <https://github.com/amrei-stammann/alpaca>

**BugReports** <https://github.com/amrei-stammann/alpaca/issues>

**RoxygenNote** 7.2.1

**Suggests** bife, car, knitr, lfe, rmarkdown

**VignetteBuilder** knitr

**Encoding** UTF-8

**NeedsCompilation** yes

**Author** Amrei Stammann [aut, cre],  
Daniel Czarnowske [aut] (<<https://orcid.org/0000-0002-0030-929X>>)

**Maintainer** Amrei Stammann <[amrei.stammann@rub.de](mailto:amrei.stammann@rub.de)>

**Repository** CRAN

**Date/Publication** 2022-08-10 15:20:02 UTC

## R topics documented:

alpaca-package	2
biasCorr	3
coef.APEs	4
coef.feglm	5
coef.summary.APEs	5
coef.summary.feglm	6
feglm	7
feglm.nb	8
feglmControl	9
fitted.feglm	11
getAPEs	11
getFEs	13
predict.feglm	14
print.APEs	15
print.feglm	15
print.summary.APEs	16
print.summary.feglm	16
simGLM	17
summary.APEs	17
summary.feglm	18
vcov.APEs	19
vcov.feglm	20
<b>Index</b>	<b>22</b>

---

alpaca-package	<i>alpaca: A package for fitting glm's with high-dimensional k-way fixed effects</i>
----------------	--------------------------------------------------------------------------------------

---

## Description

Provides a routine to partial out factors with many levels during the optimization of the log-likelihood function of the corresponding generalized linear model (glm). The package is based on the algorithm described in Stammann (2018) and is restricted to glm's that are based on maximum likelihood estimation and nonlinear. It also offers an efficient algorithm to recover estimates of the fixed effects in a post-estimation routine and includes robust and multi-way clustered standard errors. Further the package provides analytical bias corrections for binary choice models derived by Fernández-Val and Weidner (2016) and Hinz, Stammann, and Wanner (2020).

**Note:** Linear models are also beyond the scope of this package since there is already a comprehensive procedure available [felm](#).

---

biasCorr	<i>Asymptotic bias correction after fitting binary choice models with a one-/two-/three-way error component</i>
----------	-----------------------------------------------------------------------------------------------------------------

---

## Description

`biasCorr` is a post-estimation routine that can be used to substantially reduce the incidental parameter bias problem (Neyman and Scott (1948)) present in nonlinear fixed effects models (see Fernández-Val and Weidner (2018) for an overview). The command applies the analytical bias correction derived by Fernández-Val and Weidner (2016) and Hinz, Stammann, and Wanner (2020) to obtain bias-corrected estimates of the structural parameters and is currently restricted to `binomial` with one-, two-, and three-way fixed effects.

## Usage

```
biasCorr(object = NULL, L = 0L, panel.structure = c("classic", "network"))
```

## Arguments

<code>object</code>	an object of class "feglm"; currently restricted to <code>binomial</code> .
<code>L</code>	unsigned integer indicating a bandwidth for the estimation of spectral densities proposed by Hahn and Kuersteiner (2011). Default is zero, which should be used if all regressors are assumed to be strictly exogenous with respect to the idiosyncratic error term. In the presence of weakly exogenous regressors, e.g. lagged outcome variables, Fernández-Val and Weidner (2016, 2018) suggest to choose a bandwidth between one and four. Note that the order of factors to be partialled out is important for bandwidths larger than zero (see vignette for details).
<code>panel.structure</code>	a string equal to "classic" or "network" which determines the structure of the panel used. "classic" denotes panel structures where for example the same cross-sectional units are observed several times (this includes pseudo panels). "network" denotes panel structures where for example bilateral trade flows are observed for several time periods. Default is "classic".

## Value

The function `biasCorr` returns a named list of classes "biasCorr" and "feglm".

## References

- Czarnowske, D. and A. Stammann (2020). "Fixed Effects Binary Choice Models: Estimation and Inference with Long Panels". ArXiv e-prints.
- Fernández-Val, I. and M. Weidner (2016). "Individual and time effects in nonlinear panel models with large N, T". *Journal of Econometrics*, 192(1), 291-312.
- Fernández-Val, I. and M. Weidner (2018). "Fixed effects estimation of large-t panel data models". *Annual Review of Economics*, 10, 109-138.

Hahn, J. and G. Kuersteiner (2011). "Bias reduction for dynamic nonlinear panel models with fixed effects". *Econometric Theory*, 27(6), 1152-1191.

Hinz, J., A. Stammann, and J. Wanner (2020). "State Dependence and Unobserved Heterogeneity in the Extensive Margin of Trade". ArXiv e-prints.

Neyman, J. and E. L. Scott (1948). "Consistent estimates based on partially consistent observations". *Econometrica*, 16(1), 1-32.

### See Also

[feglm](#)

### Examples

```
# Generate an artificial data set for logit models
library(alpaca)
data <- simGLM(1000L, 20L, 1805L, model = "logit")

# Fit 'feglm()'
mod <- feglm(y ~ x1 + x2 + x3 | i + t, data)

# Apply analytical bias correction
mod.bc <- biasCorr(mod)
summary(mod.bc)
```

---

coef.APEs

*Extract estimates of average partial effects*

---

### Description

[coef.APEs](#) is a generic function which extracts estimates of the average partial effects from objects returned by [getAPEs](#).

### Usage

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

### Arguments

`object` an object of class "APEs".  
`...` other arguments.

### Value

The function [coef.APEs](#) returns a named vector of estimates of the average partial effects.

**See Also**[getAPEs](#)

---

coef.feglm	<i>Extract estimates of structural parameters</i>
------------	---------------------------------------------------

---

**Description**

[coef.feglm](#) is a generic function which extracts estimates of the structural parameters from objects returned by [feglm](#).

**Usage**

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

**Arguments**

object	an object of class "feglm".
...	other arguments.

**Value**

The function [coef.feglm](#) returns a named vector of estimates of the structural parameters.

**See Also**[feglm](#)

---

coef.summary.APEs	<i>Extract coefficient matrix for average partial effects</i>
-------------------	---------------------------------------------------------------

---

**Description**

[coef.summary.APEs](#) is a generic function which extracts a coefficient matrix for average partial effects from objects returned by [getAPEs](#).

**Usage**

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

**Arguments**

object	an object of class "summary.APEs".
...	other arguments.

**Value**

The function `coef.summary.APEs` returns a named matrix of estimates related to the average partial effects.

**See Also**

[getAPEs](#)

---

<code>coef.summary.feglm</code>	<i>Extract coefficient matrix for structural parameters</i>
---------------------------------	-------------------------------------------------------------

---

**Description**

`coef.summary.feglm` is a generic function which extracts a coefficient matrix for structural parameters from objects returned by `feglm`.

**Usage**

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

**Arguments**

<code>object</code>	an object of class "summary.feglm".
<code>...</code>	other arguments.

**Value**

The function `coef.summary.feglm` returns a named matrix of estimates related to the structural parameters.

**See Also**

[feglm](#)

## Description

`feglm` can be used to fit generalized linear models with many high-dimensional fixed effects. The estimation procedure is based on unconditional maximum likelihood and can be interpreted as a “weighted demeaning” approach that combines the work of Gaure (2013) and Stammann et. al. (2016). For technical details see Stammann (2018). The routine is well suited for large data sets that would be otherwise infeasible to use due to memory limitations.

**Remark:** The term fixed effect is used in econometrician’s sense of having intercepts for each level in each category.

## Usage

```
feglm(  
  formula = NULL,  
  data = NULL,  
  family = binomial(),  
  weights = NULL,  
  beta.start = NULL,  
  eta.start = NULL,  
  control = NULL  
)
```

## Arguments

formula	an object of class "formula": a symbolic description of the model to be fitted. formula must be of type $y \sim x \mid k$ , where the second part of the formula refers to factors to be concentrated out. It is also possible to pass additional variables to <code>feglm</code> (e.g. to cluster standard errors). This can be done by specifying the third part of the formula: $y \sim x \mid k \mid \text{add}$ .
data	an object of class "data.frame" containing the variables in the model.
family	a description of the error distribution and link function to be used in the model. Similar to <code>glm.fit</code> this has to be the result of a call to a family function. Default is <code>binomial()</code> . See <code>family</code> for details of family functions.
weights	an optional string with the name of the 'prior weights' variable in data.
beta.start	an optional vector of starting values for the structural parameters in the linear predictor. Default is $\beta = \mathbf{0}$ .
eta.start	an optional vector of starting values for the linear predictor.
control	a named list of parameters for controlling the fitting process. See <code>feglmControl</code> for details.

## Details

If `feglm` does not converge this is often a sign of linear dependence between one or more regressors and a fixed effects category. In this case, you should carefully inspect your model specification.

## Value

The function `feglm` returns a named list of class `"feglm"`.

## References

Gaure, S. (2013). "OLS with Multiple High Dimensional Category Variables". Computational Statistics and Data Analysis, 66.

Marschner, I. (2011). "glm2: Fitting generalized linear models with convergence problems". The R Journal, 3(2).

Stammann, A., F. Heiss, and D. McFadden (2016). "Estimating Fixed Effects Logit Models with Large Panel Data". Working paper.

Stammann, A. (2018). "Fast and Feasible Estimation of Generalized Linear Models with High-Dimensional k-Way Fixed Effects". ArXiv e-prints.

## Examples

```
# Generate an artificial data set for logit models
library(alpaca)
data <- simGLM(1000L, 20L, 1805L, model = "logit")

# Fit 'feglm()'
mod <- feglm(y ~ x1 + x2 + x3 | i + t, data)
summary(mod)
```

---

feglm.nb

*Efficiently fit negative binomial glm's with high-dimensional k-way fixed effects*

---

## Description

`feglm.nb` can be used to fit negative binomial generalized linear models with many high-dimensional fixed effects (see `feglm`).

## Usage

```
feglm.nb(
  formula = NULL,
  data = NULL,
  weights = NULL,
  beta.start = NULL,
```



```
    eta.start = NULL,  
    init.theta = NULL,  
    link = c("log", "identity", "sqrt"),  
    control = NULL  
  )
```

### Arguments

formula, data, weights, beta.start, eta.start, control  
see [feglm](#).

init.theta      an optional initial value for the theta parameter (see [glm.nb](#)).

link             the link function. Must be one of "log", "sqrt", or "identity".

### Details

If `feglm.nb` does not converge this is usually a sign of linear dependence between one or more regressors and a fixed effects category. In this case, you should carefully inspect your model specification.

### Value

The function `feglm.nb` returns a named list of class "feglm".

### References

- Gaure, S. (2013). "OLS with Multiple High Dimensional Category Variables". Computational Statistics and Data Analysis. 66.
- Marschner, I. (2011). "glm2: Fitting generalized linear models with convergence problems". The R Journal, 3(2).
- Stammann, A., F. Heiss, and D. McFadden (2016). "Estimating Fixed Effects Logit Models with Large Panel Data". Working paper.
- Stammann, A. (2018). "Fast and Feasible Estimation of Generalized Linear Models with High-Dimensional k-Way Fixed Effects". ArXiv e-prints.

### See Also

[glm.nb](#), [feglm](#)

---

feglmControl

*Set feglm Control Parameters*

---

### Description

Set and change parameters used for fitting [feglm](#).

**Note:** [feglm.control](#) is deprecated and will be removed soon.

**Usage**

```
feglmControl(
  dev.tol = 1e-08,
  center.tol = 1e-08,
  iter.max = 25L,
  limit = 10L,
  trace = FALSE,
  drop.pc = TRUE,
  keep.mx = TRUE,
  conv.tol = NULL,
  rho.tol = NULL,
  pseudo.tol = NULL,
  step.tol = NULL
)

feglm.control(...)
```

**Arguments**

<code>dev.tol</code>	tolerance level for the first stopping condition of the maximization routine. The stopping condition is based on the relative change of the deviance in iteration $r$ and can be expressed as follows: $ dev_r - dev_{r-1}  / (0.1 +  dev_r ) < tol$ . Default is $1.0e-08$ .
<code>center.tol</code>	tolerance level for the stopping condition of the centering algorithm. The stopping condition is based on the relative change of the centered variable similar to <a href="#">felm</a> . Default is $1.0e-08$ .
<code>iter.max</code>	unsigned integer indicating the maximum number of iterations in the maximization routine. Default is 25L.
<code>limit</code>	unsigned integer indicating the maximum number of iterations of <a href="#">theta.ml</a> . Default is 10L.
<code>trace</code>	logical indicating if output should be produced in each iteration. Default is FALSE.
<code>drop.pc</code>	logical indicating to drop observations that are perfectly classified/separated and hence do not contribute to the log-likelihood. This option is useful to reduce the computational costs of the maximization problem and improves the numerical stability of the algorithm. Note that dropping perfectly separated observations does not affect the estimates. Default is TRUE.
<code>keep.mx</code>	logical indicating if the centered regressor matrix should be stored. The centered regressor matrix is required for some covariance estimators, bias corrections, and average partial effects. This option saves some computation time at the cost of memory. Default is TRUE.
<code>conv.tol, rho.tol</code>	deprecated; step-halving is now similar to <a href="#">glm.fit2</a> .
<code>pseudo.tol</code>	deprecated; use <code>center.tol</code> instead.
<code>step.tol</code>	deprecated; termination conditions is now similar to <a href="#">glm</a> .
<code>...</code>	arguments passed to the deprecated function <a href="#">feglm.control</a> .

**Value**

The function `feglmControl` returns a named list of control parameters.

**See Also**

`feglm`

---

fitted.feglm	<i>Extract feglm fitted values</i>
--------------	------------------------------------

---

**Description**

`fitted.feglm` is a generic function which extracts fitted values from an object returned by `feglm`.

**Usage**

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

**Arguments**

object	an object of class "feglm".
...	other arguments.

**Value**

The function `fitted.feglm` returns a vector of fitted values.

**See Also**

`feglm`

---

getAPEs	<i>Compute average partial effects after fitting binary choice models with a one-/two-/three-way error component</i>
---------	----------------------------------------------------------------------------------------------------------------------

---

**Description**

`getAPEs` is a post-estimation routine that can be used to estimate average partial effects with respect to all covariates in the model and the corresponding covariance matrix. The estimation of the covariance is based on a linear approximation (delta method) plus an optional finite population correction. Note that the command automatically determines which of the regressors are binary or non-binary.

**Remark:** The routine currently does not allow to compute average partial effects based on functional forms like interactions and polynomials.

**Usage**

```
getAPEs(
  object = NULL,
  n.pop = NULL,
  panel.structure = c("classic", "network"),
  sampling.fe = c("independence", "unrestricted"),
  weak.exo = FALSE
)
```

**Arguments**

<code>object</code>	an object of class "biasCorr" or "feglm"; currently restricted to <a href="#">binomial</a> .
<code>n.pop</code>	unsigned integer indicating a finite population correction for the estimation of the covariance matrix of the average partial effects proposed by Cruz-Gonzalez, Fernández-Val, and Weidner (2017). The correction factor is computed as follows: $(n^* - n)/(n^* - 1)$ , where $n^*$ and $n$ are the sizes of the entire population and the full sample size. Default is NULL, which refers to a factor of zero and a covariance obtained by the delta method.
<code>panel.structure</code>	a string equal to "classic" or "network" which determines the structure of the panel used. "classic" denotes panel structures where for example the same cross-sectional units are observed several times (this includes pseudo panels). "network" denotes panel structures where for example bilateral trade flows are observed for several time periods. Default is "classic".
<code>sampling.fe</code>	a string equal to "independence" or "unrestricted" which imposes sampling assumptions about the unobserved effects. "independence" imposes that all unobserved effects are independent sequences. "unrestricted" does not impose any sampling assumptions. Note that this option only affects the optional finite population correction. Default is "independence".
<code>weak.exo</code>	logical indicating if some of the regressors are assumed to be weakly exogenous (e. g. predetermined). If object is of class "biasCorr", the option will be automatically set to TRUE if the chosen bandwidth parameter is larger than zero. Note that this option only affects the estimation of the covariance matrix. Default is FALSE, which assumes that all regressors are strictly exogenous.

**Value**

The function [getAPEs](#) returns a named list of class "APEs".

**References**

- Cruz-Gonzalez, M., I. Fernández-Val, and M. Weidner (2017). "Bias corrections for probit and logit models with two-way fixed effects". *The Stata Journal*, 17(3), 517-545.
- Czarnowske, D. and A. Stammann (2020). "Fixed Effects Binary Choice Models: Estimation and Inference with Long Panels". ArXiv e-prints.
- Fernández-Val, I. and M. Weidner (2016). "Individual and time effects in nonlinear panel models with large N, T". *Journal of Econometrics*, 192(1), 291-312.

Fernández-Val, I. and M. Weidner (2018). "Fixed effects estimation of large-t panel data models". *Annual Review of Economics*, 10, 109-138.

Hinz, J., A. Stammann, and J. Wanner (2020). "State Dependence and Unobserved Heterogeneity in the Extensive Margin of Trade". ArXiv e-prints.

Neyman, J. and E. L. Scott (1948). "Consistent estimates based on partially consistent observations". *Econometrica*, 16(1), 1-32.

### See Also

[biasCorr](#), [feglm](#)

### Examples

```
# Generate an artificial data set for logit models
library(alpaca)
data <- simGLM(1000L, 20L, 1805L, model = "logit")

# Fit 'feglm()'
mod <- feglm(y ~ x1 + x2 + x3 | i + t, data)

# Compute average partial effects
mod.ape <- getAPEs(mod)
summary(mod.ape)

# Apply analytical bias correction
mod.bc <- biasCorr(mod)
summary(mod.bc)

# Compute bias-corrected average partial effects
mod.ape.bc <- getAPEs(mod.bc)
summary(mod.ape.bc)
```

---

getFEs

*Efficiently recover estimates of the fixed effects after fitting feglm*

---

### Description

Recover estimates of the fixed effects by alternating between the normal equations of the fixed effects as shown by Stammann (2018).

**Remark:** The system might not have a unique solution since we do not take collinearity into account. If the solution is not unique, an estimable function has to be applied to our solution to get meaningful estimates of the fixed effects. See Gaure (n. d.) for an extensive treatment of this issue.

### Usage

```
getFEs(object = NULL, alpha.tol = 1e-08)
```

**Arguments**

object            an object of class "feglm".

alpha.tol        tolerance level for the stopping condition. The algorithm is stopped in iteration  $i$  if  $\|\alpha_i - \alpha_{i-1}\|_2 < tol \|\alpha_{i-1}\|_2$ . Default is  $1.0e-08$ .

**Value**

The function `getFEs` returns a named list containing named vectors of estimated fixed effects.

**References**

Gaure, S. (n. d.). "Multicollinearity, identification, and estimable functions". Unpublished.

Stammann, A. (2018). "Fast and Feasible Estimation of Generalized Linear Models with High-Dimensional k-way Fixed Effects". ArXiv e-prints.

**See Also**

[feglm](#)

---

predict.feglm	<i>Predict method for feglm fits</i>
---------------	--------------------------------------

---

**Description**

`predict.feglm` is a generic function which obtains predictions from an object returned by `feglm`.

**Usage**

```
## S3 method for class 'feglm'
predict(object, type = c("link", "response"), ...)
```

**Arguments**

object            an object of class "feglm".

type              the type of prediction required. "link" is on the scale of the linear predictor whereas "response" is on the scale of the response variable. Default is "link".

...                other arguments.

**Value**

The function `predict.feglm` returns a vector of predictions.

**See Also**

[feglm](#)

---

print.APEs	<i>Print APEs</i>
------------	-------------------

---

**Description**

[print.APEs](#) is a generic function which displays some minimal information from objects returned by [getAPEs](#).

**Usage**

```
## S3 method for class 'APEs'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

**Arguments**

x	an object of class "APEs".
digits	unsigned integer indicating the number of decimal places. Default is max(3L, getOption("digits") - 3L).
...	other arguments.

**See Also**

[getAPEs](#)

---

print.feglm	<i>Print feglm</i>
-------------	--------------------

---

**Description**

[print.feglm](#) is a generic function which displays some minimal information from objects returned by [feglm](#).

**Usage**

```
## S3 method for class 'feglm'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

**Arguments**

x	an object of class "feglm".
digits	unsigned integer indicating the number of decimal places. Default is max(3L, getOption("digits") - 3L).
...	other arguments.

**See Also**

[feglm](#)

---

```
print.summary.APEs    Print summary.APEs
```

---

**Description**

[print.summary.APEs](#) is a generic function which displays summary statistics from objects returned by [summary.APEs](#).

**Usage**

```
## S3 method for class 'summary.APEs'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

**Arguments**

x	an object of class "summary.APEs".
digits	unsigned integer indicating the number of decimal places. Default is max(3L, getOption("digits") - 3L).
...	other arguments.

**See Also**

[getAPEs](#)

---

```
print.summary.feglm  Print summary.feglm
```

---

**Description**

[print.summary.feglm](#) is a generic function which displays summary statistics from objects returned by [summary.feglm](#).

**Usage**

```
## S3 method for class 'summary.feglm'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

**Arguments**

x	an object of class "summary.feglm".
digits	unsigned integer indicating the number of decimal places. Default is max(3L, getOption("digits") - 3L).
...	other arguments.

**See Also**

[feglm](#)



---

simGLM	<i>Generate an artificial data set for some GLM's with two-way fixed effects</i>
--------	----------------------------------------------------------------------------------

---

### Description

Constructs an artificial data set with  $n$  cross-sectional units observed for  $t$  time periods for logit, poisson, or gamma models. The “true” linear predictor ( $\eta$ ) is generated as follows:

$$\eta_{it} = \mathbf{x}'_{it}\boldsymbol{\beta} + \alpha_i + \gamma_t,$$

where  $\mathbf{X}$  consists of three independent standard normally distributed regressors. Both parameter referring to the unobserved heterogeneity ( $\alpha_i$  and  $\gamma_t$ ) are generated as iid. standard normal and the structural parameters are set to  $\boldsymbol{\beta} = [1, -1, 1]'$ .

**Note:** The poisson and gamma model are based on the logarithmic link function.

### Usage

```
simGLM(n = NULL, t = NULL, seed = NULL, model = c("logit", "poisson", "gamma"))
```

### Arguments

n	a strictly positive integer equal to the number of cross-sectional units.
t	a strictly positive integer equal to the number of time periods.
seed	a seed to ensure reproducibility.
model	a string equal to "logit", "poisson", or "gamma". Default is "logit".

### Value

The function `simGLM` returns a data.frame with 6 variables.

### See Also

[feglm](#)

---

summary.APEs	<i>Summarizing models of class APEs</i>
--------------	-----------------------------------------

---

### Description

Summary statistics for objects of class "APEs".

### Usage

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

**Arguments**

object            an object of class "APEs".  
 ...              other arguments.

**Value**

Returns an object of class "summary.APEs" which is a list of summary statistics of object.

**See Also**

[getAPEs](#)

---

summary.feglm	<i>Summarizing models of class feglm</i>
---------------	------------------------------------------

---

**Description**

Summary statistics for objects of class "feglm".

**Usage**

```
## S3 method for class 'feglm'
summary(
  object,
  type = c("hessian", "outer.product", "sandwich", "clustered"),
  cluster = NULL,
  cluster.vars = NULL,
  ...
)
```

**Arguments**

object            an object of class "feglm".  
 type             the type of covariance estimate required. "hessian" refers to the inverse of the negative expected Hessian after convergence and is the default option. "outer.product" is the outer-product-of-the-gradient estimator, "sandwich" is the sandwich estimator (sometimes also referred as robust estimator), and "clustered" computes a clustered covariance matrix given some cluster variables.  
 cluster          a symbolic description indicating the clustering of observations.  
 cluster.vars    deprecated; use cluster instead.  
 ...              other arguments.

**Details**

Multi-way clustering is done using the algorithm of Cameron, Gelbach, and Miller (2011). An example is provided in the vignette "Replicating an Empirical Example of International Trade".

**Value**

Returns an object of class "summary.feglm" which is a list of summary statistics of object.

**References**

Cameron, C., J. Gelbach, and D. Miller (2011). "Robust Inference With Multiway Clustering". *Journal of Business & Economic Statistics* 29(2).

**See Also**

[feglm](#)

---

vcov.APEs

*Compute covariance matrix after estimating APEs*

---

**Description**

[vcov.APEs](#) estimates the covariance matrix for the estimator of the average partial effects from objects returned by [getAPEs](#).

**Usage**

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

**Arguments**

object            an object of class "APEs".  
...                other arguments.

**Value**

The function [vcov.APEs](#) returns a named matrix of covariance estimates.

**See Also**

[getAPEs](#)

vcov.feglm

*Compute covariance matrix after fitting feglm***Description**

`vcov.feglm` estimates the covariance matrix for the estimator of the structural parameters from objects returned by `feglm`. The covariance is computed from the Hessian, the scores, or a combination of both after convergence.

**Usage**

```
## S3 method for class 'feglm'
vcov(
  object,
  type = c("hessian", "outer.product", "sandwich", "clustered"),
  cluster = NULL,
  cluster.vars = NULL,
  ...
)
```

**Arguments**

<code>object</code>	an object of class "feglm".
<code>type</code>	the type of covariance estimate required. "hessian" refers to the inverse of the negative expected Hessian after convergence and is the default option. "outer.product" is the outer-product-of-the-gradient estimator, "sandwich" is the sandwich estimator (sometimes also referred as robust estimator), and "clustered" computes a clustered covariance matrix given some cluster variables.
<code>cluster</code>	a symbolic description indicating the clustering of observations.
<code>cluster.vars</code>	deprecated; use <code>cluster</code> instead.
<code>...</code>	other arguments.

**Details**

Multi-way clustering is done using the algorithm of Cameron, Gelbach, and Miller (2011). An example is provided in the vignette "Replicating an Empirical Example of International Trade".

**Value**

The function `vcov.feglm` returns a named matrix of covariance estimates.

**References**

Cameron, C., J. Gelbach, and D. Miller (2011). "Robust Inference With Multiway Clustering". *Journal of Business & Economic Statistics* 29(2).

*vcov.feglm*

21

**See Also**

[feglm](#)

# Index

alpaca-package, [2](#)

biasCorr, [3](#), [3](#), [13](#)  
binomial, [3](#), [12](#)

coef.APEs, [4](#), [4](#)  
coef.feglm, [5](#), [5](#)  
coef.summary.APEs, [5](#), [5](#), [6](#)  
coef.summary.feglm, [6](#), [6](#)

family, [7](#)  
feglm, [4–7](#), [7](#), [8](#), [9](#), [11](#), [13–17](#), [19–21](#)  
feglm.control, [9](#), [10](#)  
feglm.control (feglmControl), [9](#)  
feglm.nb, [8](#)  
feglmControl, [7](#), [9](#), [11](#)  
felm, [2](#), [10](#)  
fitted.feglm, [11](#), [11](#)

getAPEs, [4–6](#), [11](#), [11](#), [12](#), [15](#), [16](#), [18](#), [19](#)  
getFES, [13](#), [14](#)  
glm, [10](#)  
glm.fit, [7](#)  
glm.nb, [9](#)

predict.feglm, [14](#), [14](#)  
print.APEs, [15](#), [15](#)  
print.feglm, [15](#), [15](#)  
print.summary.APEs, [16](#), [16](#)  
print.summary.feglm, [16](#), [16](#)

simGLM, [17](#), [17](#)  
summary.APEs, [16](#), [17](#)  
summary.feglm, [16](#), [18](#)

theta.ml, [10](#)

vcov.APEs, [19](#), [19](#)  
vcov.feglm, [20](#), [20](#)