Package 'effClust'

July 22, 2025

Title Calculate Effective Number of Clusters for a Linear Model

Version 0.8.0

Description Calculates the (approximate) effective number of clusters for a regression model, as described in Carter, Schnepel, and Steigerwald (2017) <doi:10.1162/REST_a_00639>. The effective number of clusters is a statistic to assess the reliability of asymptotic inference when sampling or treatment assignment is clustered. Methods are implemented for stats::lm(), plm::plm(), and fixest::feols(). There is also a formula method.

Depends R (>= 3.5.0)

Suggests plm, data.table

Imports fixest, stats

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.1

NeedsCompilation no

Author Joe Ritter [aut, cre]

Maintainer Joe Ritter <R-effclust@proton.me>

Repository CRAN

Date/Publication 2024-03-06 14:30:02 UTC

Contents

`effClust.default` *Compute Approximate Effective Number of Clusters for Regression Coefficients*

Description

Specifically, for each coefficient the function returns the quantity G^{*A}, the feasible version of G^*, introduced by Carter, Schnepel, and Steigerwald (2017). G^{*A} does not accommodate multi-way clustering.

Usage

```
## Default S3 method:
effClust(
  object,
  cluster,
  tags = colnames(object),
  rho = 0.999,
  nominal = FALSE,
  XpXinv = NULL,
  ...
)

## S3 method for class 'fixest'
effClust(
  object,
  cluster,
  include.only = NULL,
  exclude = NULL,
  fixed = FALSE,
  nominal = FALSE,
  rho = 0.999,
  ...
)

## S3 method for class 'formula'
effClust(
  object,
  cluster,
  data,
  subset = NULL,
  include.only = NULL,
  exclude = NULL,
  fixed = FALSE,
  nominal = FALSE,
  rho = 0.999,
  ...
```

```
)

## S3 method for class 'lm'
effClust(
  object,
  cluster,
  include.only = NULL,
  exclude = NULL,
  fixed = FALSE,
  nominal = FALSE,
  rho = 0.999,
  ...
)

## S3 method for class 'plm'
effClust(
  object,
  cluster,
  include.only = NULL,
  exclude = NULL,
  fixed = FALSE,
  nominal = FALSE,
  rho = 0.999,
  ...
)

effClust(object, cluster, ...)
```

Arguments

object	a formula or an object of class "lm", "plm", "fixest" (with method "feols"), or (for the default method) a matrix of regressors
cluster	cluster identifier: a variable, a formula (see details), or, a length one character vector naming a variable in data.
tags	a character vector containing a subset of the column names of object
rho	numeric scalar that changes assumed variance matrix (see details)
nominal	logical indicating whether the number of clusters should be included as the first element of the return vector
XpXinv	the $k \times k$ matrix $(X'X)^{-1}$ where X is $n \times k$ matrix object for the default method
...	arguments passed to methods
include.only	a vector of regular expressions for variables to be included in the return (implying others excluded)
exclude	a vector of regular expressions for variables to be excluded from the return
fixed	logical indicates how regular expressions in exclude or include.only should be evaluated (as in grep)

data	a dataframe (see details)
subset	an optional vector used to select cases from data

Details

When object is a formula, it does not need a response variable (left-hand side), but if the response variable might have different missing cases than the right-hand side, it should be included.

Cluster fixed effects, if any, may be explicitly included in the main formula, or be specified with the syntax used by fixest:

~ X1 + X2 + factor(id) (fixed effects id in main formula)

~ X1 + X2 | id (fixest way to specify fixed effects)

The first approach can be computationally demanding if there are too many units. Alternatively, data can contain appropriately transformed variables (see the example below).

For regression objects, when cluster is a formula or a one element character vector, it is evaluated in the context of the data used for the model, and the data and subset arguments are ignored.

The data argument is required when object is a formula. In this case, object is evaluated in the context of the data argument. If cluster is a formula or length 1 character vector, it is interpreted as the name of a column of data.

If cluster is a variable, its length must equal nobs(object) for regression objects or the number of rows in data when object is a formula. If cluster contains NA, a warning is issued.

By default G^{*A} is returned for all coefficients. The output may be limited by using include.only or exclude. This does not affect the computation for coefficients that are included, only the vector returned.

The regular expressions used by include.only or exclude should refer to the contents of names(coef(object)), which might differ from how the regression formula refers to the same things. For example, groups of variables entered by using something like factor(statefip) in the regression formula show up in the coefficient vector names looking like factor(statefip)27. They can be excluded en mass by the regular expression exclude="^factor\(statefip\)" (parentheses need to be escaped in a regular expression). Alternatively, you can specify fixed=TRUE, which has the same meaning as for grep.

Carter et al. recommend assuming every element of the variance matrix of errors in a cluster is 1, a conservative scenario. However, as explained in "A Note on Computation," the default rho is 0.999, so that the variance matrix has 1 on the diagonal and 0.999 off the diagonal. It is possible to set rho=1, but that produces incorrect values in when there are cluster fixed effects. In other cases the computation is not very sensitive to the value of rho, so there is no appreciable difference between rho=1 and rho=0.999. MacKinnon, Nielsen, and Webb (2022) argue that cluster fixed effects usually remove much of the intra-cluster correlation, justifying rho close to zero. (However, this is inconsistent with the "worst case scenario" approach that motivates Carter et al.'s recommendation.)

effClust will return a value for a glm object (or a formula intended for a GLM). The result might or might not be a useful diagnostic. On one hand, G^* is fundamentally about the characteristics of the clusters. On the other hand, the derivation by Carter et. al. is based on linear regression.

Default method. The default method is a bare-bones function that does the actual calculation of the effective number of clusters; it is mainly intended to be a tool for adding methods. It provides none of the error checking that is normally performed by the effClust methods.

The matrix `object` must have column names. It also must include a column of ones if the hypothetical regression includes an intercept.

If `XpXinv` is provided, it will be indexed using `tags`, so its row and column names must be identical to (or a superset of) `tags`.

If `object` cannot be coerced to numeric, the function will fail. Logical columns of `object` will be coerced to numeric, but factors will *not* be expanded to dummy variables as done by `model.matrix`.

Value

vector of effective number of clusters for coefficients.

References

Andrew V. Carter, Kevin T. Schnepel, and Douglas G. Steigerwald, "Asymptotic Behavior of a t-test Robust to Cluster Heterogeneity," *The Review of Economics and Statistics,* October 2017, 99(4). doi:10.1162/REST_a_00639.

James G. MacKinnon, Morten Ørregaard Nielsen, and Matthew D. Webb, "Leverage, Influence, and the Jackknife in Clustered Regression Models: Reliable Inference Using summclust," QED Working Paper 1483, Queen's University (2022). https://www.econ.queensu.ca/research/working-papers/1483.

Examples

```
# some data with correlated errors
set.seed(85914270)
G <- 50
cl.sizes <- sample(10:100, G, replace=TRUE)
n <- sum(cl.sizes)
id <- rep(1:G, cl.sizes)
X1 <- rchisq(n, 5)
X2 <- ifelse(id %% 4 == 0, 1, 0)
e <- rnorm(n)
eg <- rep(rnorm(G), cl.sizes)
Y <- 1 + 2*X1 + 3*X2 - 1*X1*X2 + e + eg
d <- data.frame(Y, X1, X2, id)

f1 <- Y ~ X1 + X1:X2 + factor(id)
f2 <- Y ~ X1 + X1:X2 | id  # fixest syntax

r <- lm(f1, data=d)

effClust(r, ~d$id, exclude=c("factor\\(id\\)","Intercept"))
effClust(f2, ~id, data=d)

library(data.table)
setDT(d)
d[ , `:=`(X1dot = X1 - mean(X1),
          X2dot = X2 - mean(X2),
          X1X2dot = X1*X2 - mean(X1*X2)),
  by = id]
```

```
effClust(~ -1+X1dot+X1X2dot, cluster=~id, data=d)
```

Index