

# Package ‘speedglm’

January 26, 2012

**Type** Package

**Title** Fitting Linear and Generalized Linear Models to large data sets.

**Version** 0.1

**Date** 2009-09-10

**Author** Marco ENEA

**Maintainer** Marco ENEA <enea@dssm.unipa.it>

**Depends** Matrix

**Suggests** biglm

**Description** Fitting LMs and GLMs to large data sets by updating algorithms.

**License** GPL

**LazyLoad** yes

**Repository** CRAN

**Date/Publication** 2012-01-26 10:44:47

## R topics documented:

speedglm-package . . . . .	2
control . . . . .	2
speedglm . . . . .	5
speedlm . . . . .	9
summary.speedglm . . . . .	12
summary.speedlm . . . . .	14

<b>Index</b>	<b>16</b>
--------------	-----------

---

speedglm-package

*Fitting Linear and Generalized Linear Models to Large Data Sets.*

---

### Description

Fits LMs and GLMs to large data sets. For data loaded in R memory the fitting is usually fast, especially if R is linked against an optimized BLAS. For data sets of size greater of R memory, the fitting is made by an updating algorithm.

### Details

Package: speedglm  
Type: Package  
Version: 1.0  
Date: 2009-09-10  
Depends: Matrix  
License: GPL  
LazyLoad: yes

### Author(s)

Marco ENEA

Maintainer: Marco ENEA enea@dssm.unipa.it (thanks to Vito M.R. Muggeo for his helpful advice)

---

control

*Miscellanea of functions*

---

### Description

Utility functions for least squares estimation in large data sets.

### Usage

```
control(B, symmetric = TRUE, tol.values = 1e-7, tol.vectors = 1e-7, out.B = TRUE, method = "eigen")  
cp(X, w = NULL, row.chunk = NULL, sparse = FALSE)  
is.sparse(X, sparselim = .9, camp = .05)
```

**Arguments**

B	a squared matrix.
symmetric	logical, is B symmetric?
tol.values	tolerance to be consider eigenvalues equals to zero.
tol.vectors	tolerance to be consider eigenvectors equals to zero.
out.B	Have the matrix B to be returned?
method	the method to check for singularity. By default is "eigen", and an eigendecomposition of $X'X$ is made. The "Cholesky" method is faster than "eigen" and does not use tolerance, but the former seems to be more stable for opportune tolerance values.
X	the model matrix.
w	a weights vector.
sparse	logical, is X sparse?
sparselim	a real in the interval [0; 1]. It indicates the minimal proportion of zeroes in the data matrix X in order to consider X as sparse eigendec Logical. Do you want to investigate on rank of X? You may set to
row.chunk	an integer which indicates the total rows number compounding each of the first g-1 blocks. If row.chunk is not a divisor of nrow(X), the g-th block will be formed by the remaining data.
camp	the sample proportion of elements of X on which the survey will be based.

**Details**

Function `control` makes an eigendecomposition of B according established values of tolerance. Function `cp` makes the cross-product  $X'X$  by partitioning X in row-blocks. When an optimized BLAS, such as ATLAS, is not installed, the function represents an attempt to speed up the calculation and avoid overflows with medium-large data sets loaded in R memory. The results depending on processor type. Good results are obtained, for example, with an AMD Athlon dual core 1.5 Gb RAM by setting `row.chunk` to some value less than 1000. Try the example below by changing the matrix size and the value of `row.chunk`. If the matrix X is sparse, it will have class "dgCMatrix" (the package Matrix is required) and the cross-product will be made without partitioning. However, good performances are usually obtained with a very high zeroes proportion. Function `is.sparse` makes a quick sample survey on sample proportion of zeroes in X.

**Value**

for the function `control`, a list with the following elements:

XTX	the matrix product B without singularities (if there are).
rank	the rank of B
pivot	an ordered set of column indeces of B with, if the case, the last $rank + 1, \dots, p$ columns which indicate possible linear combinations.

for the function `cp`:

new.B	the matrix product $X'X$ (weighted, if w is given).
-------	---

for the function `is.sparse`:

`sparse`                    a logical value which indicates if the sample proportion of zeroes is greater than `sparselim`, with the sample proportion as attribute.

### Author(s)

Marco ENEA

### See Also

[eigen](#), [chol](#), [qr](#), [crossprod](#)

### Examples

```
#### example 1.

n <- 100000
k <- 100
x <- round(matrix(rnorm(n*k),n,k),digits=4)
y <- rnorm(n)

# if an optimized BLAS is not installed, depending on processor type, cp() may be
# faster than crossprod() for large matrices.

system.time(a1 <- crossprod(x))
system.time(a2 <- cp(x,,row.chunk = 500))
all.equal(a1, a2)

#### example 2.1.
n <- 100000
k <- 10
x <- matrix(rnorm(n*k),n,k)
x[,2] <- x[,1] + 2*x[,3] # x has rank 9
y <- rnorm(n)

# estimation by least squares
A <- function(){
  A1 <- control(crossprod(x))
  ok <- A1$pivot[1:A1$rank]
  as.vector(solve(A1$XTX,crossprod(x[,ok],y)))
}
# estimation by QR decomposition
B <- function(){
  B1 <- qr(x)
  qr.solve(x[,B1$pivot[1:B1$rank]],y)
}
system.time(a <- A())
system.time(b <- B())

all.equal(a,b)

### example 2.2
```

```
x <- matrix(c(1:5, (1:5)^2), 5, 2)
x <- cbind(x, x[, 1] + 3*x[, 2])
m <- crossprod(x)
qr(m)$rank # is 2, as it should be
control(m,method="eigen")$rank # is 2, as it should be
control(m,method="Cholesky")$rank # is wrong
```

```
### example 3.
n <- 10000
fat1 <- gl(20,500)
y <- rnorm(n)
da <- data.frame(y,fat1)
m <- model.matrix(y ~ factor(fat1),data = da)
is.sparse(m)
```

---

speedglm

*Fitting Generalized Linear Models for Large Data Sets*


---

## Description

speedglm and speedglm.wfit fit GLMs to medium-large data sets, that is those storable into the R memory. The highest performances, in terms of computation time, are obtained when R is linked against an optimized BLAS, such as ATLAS. The function shglm is for a data set stored into a file of size greater than the available memory, and takes as argument a function to manipulate connections.

## Usage

```
## S3 method for class 'data.frame':
speedglm(formula, data, family = gaussian(), weights = NULL, start = NULL,
          etastart = NULL, mustart = NULL, offset = NULL, maxit = 25, k = 2,
          sparse = NULL, set.default = list(), ...)

## S3 method for class 'matrix':
speedglm.wfit(y, X, intercept = TRUE, weights = NULL, row.chunk = NULL,
              family = gaussian(), start = NULL, etastart = NULL, mustart = NULL,
              offset = NULL, acc = 1e-08, maxit = 25, k = 2, sparselim = .9,
              camp = .01, eigendec = TRUE, tol.values = 1e-7, tol.vectors = 1e-7,
              tol.solve = .Machine$double.eps, method = "eigen", sparse = NULL,...)

## S3 method for class 'function':
shglm(formula, data, family = gaussian(), weights.fo = NULL, start = NULL,
       etastart = NULL, mustart = NULL, offset = NULL, maxit = 25, k = 2,
       chunksize = 5000, sparse = NULL, all.levels = FALSE, set.default = list(),...)
```

**Arguments**

Most of arguments are the same of `glm` or `bigglm` but with some difference.

	the same of <code>glm</code> .
<code>data</code>	as in <code>bigglm</code> , can be either a data frame or a function which uses connections. The latter case is only for <code>shglm</code> .
<code>family</code>	the same of <code>glm</code> , but it must be specified with brackets.
<code>start</code>	the same of <code>glm</code> .
<code>weights</code>	the same of <code>glm</code> , but it must be specified as <code>data\$weights</code> .
<code>weights.fo</code>	weights for the response. It must be specified as a formula (see the example below).
<code>etastart</code>	the same of <code>glm</code> .
<code>mustart</code>	the same of <code>glm</code> .
<code>offset</code>	the same of <code>glm</code> .
<code>intercept</code>	the same of <code>glm</code> .
<code>X</code>	the same of <code>x</code> in <code>glm.fit</code> .
<code>y</code>	the same of <code>glm</code> .
<code>maxit</code>	the same of <code>glm</code> .
<code>k</code>	numeric, the penalty per parameter to be used; the default <code>k = 2</code> is the classical AIC.
<code>sparse</code>	logical. Is the model matrix sparse? By default is <code>NULL</code> , so a quickly sample survey will be made.
<code>chunksize</code>	an integer indicates the number of rows of the data file to read at time.
<code>all.levels</code>	logical, are all factor's levels present in each data chunk?
<code>set.default</code>	a list in which to specify the below parameters.
<code>sparselim</code>	a real in the interval $[0, 1]$ . It indicates the minimal proportion of zeroes in the data matrix <code>X</code> in order to consider <code>X</code> as sparse.
<code>camp</code>	see the function <code>is.sparse</code> .
<code>eigendec</code>	logical. Do you want to check the rank of <code>X</code> ? You may set it to <code>false</code> if you are sure that <code>X</code> is full rank.
<code>row.chunk</code>	an integer, see the function <code>cp</code> for details.
<code>acc</code>	tolerance to be used for the estimation.
<code>tol.solve</code>	see the function <code>solve</code> .
<code>tol.values</code>	see the function <code>control</code> .
<code>tol.vectors</code>	see the function <code>control</code> .
<code>method</code>	see the function <code>control</code> .
<code>...</code>	further optional arguments.

**Details**

The function `shglm` works like `biglm`, but it checks for singularity and does not impose restrictions on factors. Since during the IWLS estimation `shglm` uses repeated accesses to data file stored, for example, into the hard disk, the estimation time could be very long. Unlike from `glm` or `biglm`, the functions of class `'speedglm'` do not use the QR decomposition, but directly solve the equations in the form of Iterative(-ly) (Re-)Weighted Least Squares (IWLS). The memory size of an object of class `'speedglm'` is  $O(p^2)$ , where  $p$  is the number of covariates. If an optimized BLAS is not installed, an attempt to speed up calculations might be done by setting `row.chunk` to some value, usually less than 1000, in `set.default`. See the function `cp` for details. If the model matrix is (very) sparse, the package `Matrix` could be used.

**Value**

<code>coefficients</code>	the estimated coefficients.
<code>logLik</code>	the log likelihood of the fitted model.
<code>iter</code>	the number of iterations of IWLS used.
<code>tol</code>	the maximal value of tolerance reached.
<code>convergence</code>	a logical value which indicates if convergence was reached.
<code>family</code>	the family object used.
<code>link</code>	the link function used.
<code>df</code>	the degrees of freedom of the model.
<code>XTX</code>	the product $X'X$ (weighted, if the case).
<code>dispersion</code>	the estimated dispersion parameter of the model.
<code>ok</code>	the set of column indices of the model matrix where the model has been fitted.
<code>rank</code>	the rank of the model matrix.
<code>RSS</code>	the estimated residual sum of squares of the fitted model.
<code>aic</code>	the estimated Akaike Information Criterion.
<code>sparse</code>	a logical value which indicates if the model matrix is sparse.
<code>deviance</code>	the estimated deviance of the fitted model.
<code>nulldf</code>	the degrees of freedom of the null model.
<code>nulldev</code>	the estimated deviance of the null model.
<code>ngoodobs</code>	the number of non-zero weighted observations.
<code>n</code>	the number of observations.
<code>intercept</code>	a logical value which indicates if an intercept has been used.
<code>terms</code>	the terms object used.
<code>call</code>	the matched call.

**Note**

All the above functions make an object of class `'speedglm'`

**Author(s)**

Marco ENEA

**References**

Enea, M. (2009) Fitting Linear Models and Generalized Linear Models with large data sets in R. In *book of short papers, conference on "Statistical Methods for the analysis of large data-sets"*, Italian Statistical Society, Chieti-Pescara, 23-25 September 2009, 411-414.

Bates, D. (2009) Comparing Least Square Calculations. Technical report. Available at <http://cran.rakanu.com/web/packages/Matrix/vignettes/Comparisons.pdf>

Lumley, T. (2009) biglm: bounded memory linear and generalized linear models. *R package version 0.7*. <http://CRAN.R-project.org/package=biglm>.

**See Also**

[speedlm](#), [bigglm](#), [glm](#)

**Examples**

```
# Not Run: the example may take a long time.
# The following comparison among glm(), bigglm() and speedglm() cannot be considered rigorous
# and exhaustive, but it is only to give an idea of the computation time.
require(biglm)
n<-50000
k<-80
y <- rgamma(n,1.5,1)
x <-round( matrix(rnorm(n*k),n,k),digits=3)
colnames(x) <-paste("s",1:k,sep = "")
da<- data.frame(y,x)
fo <- as.formula(paste("y~",paste(paste("s",1:k,sep=""),collapse="+"))))

system.time(m1 <- glm(fo,data=da,family=Gamma(log)))
system.time(m2 <- bigglm(fo,data=da,family=Gamma(log)))
system.time(m3 <- speedglm(fo,data=da,family=Gamma(log)))

# You may also try speedglm when R is linked against an optimized BLAS, otherwise try to run the following
# function. In some computers, it is faster for large data sets.
system.time(m4 <- speedglm(fo,data=da,family=Gamma(log),set.default=list(row.chunk=1000)))

#####

## The function from the bigglm's help page
make.data<-function(urlname, chunksize,...){
  conn<-NULL
  function(reset=FALSE){
    if(reset){
      if(!is.null(conn)) close(conn)

```

```

        conn<-url(urlname,open="r")
    } else{
        rval<-read.table(conn, nrows=chunksize,...)
        if (nrow(rval)==0) {
            close(conn)
            conn<-NULL
            rval<-NULL
        }
        return(rval)
    }
}

require(biglm)
da<-make.data("http://dssm.unipa.it/enea/data1.txt",
             chunksize=50,col.names=c("y","fat1","x1","x2"))

# fat1 is a factor with four levels
b1<-shglm(y~factor(fat1)+x1,weights=~I(x2^2),data=da,family=Gamma(log))
b2<-bigglm(y~factor(fat1)+x1,weights=~I(x2^2),data=da,family=Gamma(log))
summary(b1)
summary(b2)

```

---

speedlm

*Fitting Linear Models to Large Data Sets*


---

## Description

The functions of class 'speedlm' may speed up the fitting of LMs to large data sets. High performances can be obtained especially if R is linked against an optimized BLAS, such as ATLAS.

## Usage

```

# S3 method of class 'data.frame'
speedlm(formula,data,weights=NULL,offset=NULL,sparse=NULL,set.default=list(),...)

# S3 method of class 'matrix'
speedlm.fit(y,X,intercept=FALSE,offset=NULL,row.chunk=NULL,sparselim=.9,camp=.01,
           eigendec=TRUE,tol.solve=.Machine$double.eps,sparse=NULL,
           tol.values=1e-7,tol.vectors=1e-7, method = "eigen",...)

speedlm.wfit(y,X,w,intercept=FALSE,offset=NULL,row.chunk=NULL,sparselim=.9,camp=.01,
            eigendec=TRUE,tol.solve=.Machine$double.eps,sparse=NULL,
            tol.values=1e-7,tol.vectors=1e-7, method = "eigen",...)

# S3 method of class 'speedlm' (object) and 'data.frame' (data)
update.speedlm(object,data,weights=NULL,offset=NULL,sparse=NULL,all.levels=FALSE,
               set.default=list(),...)

```

**Arguments**

	Most of arguments are the same of functions <a href="#">lm</a> but with some difference.
	the same of function <a href="#">lm</a> .
<code>formula</code>	the same of function <a href="#">lm</a> , but it must always specified.
<code>weights</code>	the same of function <a href="#">lm</a> , but it must be specified as <code>data\$weights</code> .
<code>w</code>	the same of <code>weights</code> .
<code>intercept</code>	a logical value which indicates if an intercept is used.
<code>offset</code>	the same of function <a href="#">lm</a> .
<code>X</code>	the same of <code>x</code> in function <a href="#">lm</a> .
<code>y</code>	the same of function <a href="#">lm</a> .
<code>sparse</code>	logical. Is the model matrix sparse? By default is <code>NULL</code> , so a quickly sample survey will be made.
<code>set.default</code>	a list in which to specify the parameters to pass to the functions <a href="#">cp</a> , <a href="#">control</a> and <a href="#">is.sparse</a> .
<code>sparselim</code>	a value in the interval $[0, 1]$ . It indicates the minimal proportion of zeroes, in the model matrix <code>X</code> , in order to consider <code>X</code> as sparse.
<code>camp</code>	see function <a href="#">is.sparse</a> .
<code>eigendec</code>	logical. Do you want to investigate on rank of <code>X</code> ? You may set it to <code>false</code> if you are sure that <code>X</code> is full rank.
<code>row.chunk</code>	an integer, see the function <a href="#">cp</a> for details.
<code>tol.solve</code>	see function <a href="#">solve</a> .
<code>tol.values</code>	see function <a href="#">control</a> .
<code>tol.vectors</code>	see function <a href="#">control</a> .
<code>method</code>	see function <a href="#">control</a> .
<code>object</code>	an object of class <code>'speedlm'</code> .
<code>all.levels</code>	are all levels of eventual factors present in each data chunk? If so, set <code>all.levels</code> to <code>true</code> to speed up the fitting.
<code>...</code>	further optional arguments.

**Details**

Unlikely from the functions [lm](#) or [biglm](#), the functions of class `'speedlm'` do not use the QR decomposition but directly solve the normal equations. In some extreme case, this might have some problem of numerical stability but may take advantage from the use of an optimized BLAS. The memory size of an object of class `'speedlm'` is  $O(p^2)$ , where  $p$  is the number of covariates. If an optimized BLAS library is not installed, an attempt to speed up calculations may be done by setting `row.chunk` to some value, usually less than 1000, in `set.default`. See the function [cp](#) for details. Factors are permitted without limitations.

**Value**

coefficients	the estimated coefficients.
df.residual	the residual degrees of freedom.
XTX	the product $X'X$ (weighted, if the case).
A	the product $X'X$ (weighted, if the case) not checked for singularity.
Xy	the product $X'y$ (weighted, if the case).
ok	the set of column indices of the model matrix where the model has been fitted.
rank	the numeric rank of the fitted linear model.
pivot	see the function <a href="#">control</a> .
RSS	the estimated residual sums of squares of the fitted model.
sparse	a logical value indicating if the model matrix is sparse.
deviance	the estimated deviance of the fitted model.
weights	the weights used in the last updating.
zero.w	the number of non-zero weighted observations.
n.obs	the number of observations.
nvar	the number of independent variables.
terms	the terms object used.
intercept	a logical value which indicates if an intercept has been used.
call	the matched call.
...	others values necessary to update the estimation.

**Note**

All the above functions make an object of class 'speedlm'.

**Author(s)**

Marco ENEA

**References**

Enea, M. (2009) Fitting Linear Models and Generalized Linear Models With Large Data Sets in R. In *book of short papers, conference on "Statistical Methods for the analysis of large data-sets"*, Italian Statistical Society, Chieti-Pescara, 23-25 September 2009, 411-414.

Klotz, J.H. (1995) Updating Simple Linear Regression. *Statistica Sinica*, **5**, 399-403.

Bates, D. (2009) Comparing Least Square Calculations. Technical report. Available at <http://cran.rakanu.com/web/packages/Matrix/vignettes/Comparisons.pdf>

Lumley, T. (2009) biglm: bounded memory linear and generalized linear models. *R package version 0.7* <http://CRAN.R-project.org/package=biglm>.

**See Also**

[summary.speedlm](#), [speedglm](#), [lm](#), and [biglm](#)

**Examples**

```
n <- 1000
k <- 3
y <- rnorm(n)
x <- round(matrix(rnorm(n * k), n, k), digits = 3)
colnames(x) <- c("s1", "s2", "s3")
da <- data.frame(y, x)
do1 <- da[1:300,]
do2 <- da[301:700,]
do3 <- da[701:1000,]

m1 <- speedlm(y ~ s1 + s2 + s3, data = do1)
m1 <- update(m1, data = do2)
m1 <- update(m1, data = do3)

m2 <- lm(y ~ s1 + s2 + s3, data = da)
summary(m1)
summary(m2)
```

---

summary.speedglm

*Methods to summarize Generalized Linear Models fits*

---

**Description**

summary method for the class 'speedglm'.

**Usage**

```
## S3 method for class 'speedglm'
summary(object, correlation=FALSE, ...)
## S3 method for class 'speedglm'
coef(object, ...)
## S3 method for class 'speedglm'
vcov(object, ...)
## S3 method for class 'speedglm'
logLik(object, ...)
## S3 method for class 'speedglm'
AIC(object, ...)
```

**Arguments**

object	an object of class 'speedlm'.
correlation	logical. Do you want to print the correlation matrix? By default it is false.
...	further optional arguments

**Value**

coefficients	the matrix of coefficients, standard errors, z-statistics and two-side p-values.
df.residual	the component from object.
df.null	the component from object.
null.deviance	the component from object.
deviance	the component from object.
family	the component from object.
call	the component from object.
AIC	the Akaike Information Criterion.
RSS	Residuals sums of squares.
correlation	(only if correlation is true.) The correlations of the estimated coefficients.
logLik	the log-likelihood value.
rank	the component from object.
dispersion	the estimated dispersion parameter of the fitted model.
convergence	the component from object.
iter	the component from object.
tol	the component from object.

**Author(s)**

Marco ENEA

**See Also**

[speedglm](#)

**Examples**

```
n<-1000
k<-5
y <- rgamma(n,1.5,1)
x <-round( matrix(rnorm(n*k),n,k),digits=3)
colnames(x) <-paste("s",1:k,sep = "")
da<- data.frame(y,x)
fo <- as.formula(paste("y~",paste(paste("s",1:k,sep=""),collapse="+")))

m4 <- speedglm(fo,data=da,family=Gamma(log))
summary(m4)
```

---

summary.speedlm            *Methods to summarize Linear Models fits*

---

## Description

summary method for class 'speedlm'.

## Usage

```
## S3 method for class 'speedlm'
summary(object, correlation = FALSE, ...)
## S3 method for class 'speedlm'
coef(object, ...)
## S3 method for class 'speedlm'
vcov(object, ...)
## S3 method for class 'speedlm'
logLik(object, ...)
## S3 method for class 'speedlm'
AIC(object, ..., k = 2)
```

## Arguments

object	an object of class 'speedlm'.
correlation	logical. Do you want to print the correlation matrix? By default it is false.
k	numeric, the penalty per parameter to be used; the default $k = 2$ is the classical AIC.
...	further optional arguments

## Value

coefficients	the matrix of coefficients, standard errors, t-statistics and two-side p-values.
rdf	degrees of freedom of the fitted model. It is a component from object.
call	the component from object.
r.squared	$R^2$ , the fraction of variance explained by the model.
adj.r.squared	the "adjusted" $R^2$ statistic, penalizing for higher p.
fstatistic	(for models including non-intercept terms) a 3-vector with the value of the F-statistic with its numerator and denominator degrees of freedom.
f.pvalue	p-value of the F-statistic.
RSS	Residual sum of squares.
var.res	estimated variance of residuals.
rank	the component from object.
correlation	(only if correlation is true) the correlations of the estimated parameters.
...	the results from the functions logLik, AIC and vcov.

**Author(s)**

Marco ENEA

**See Also**

[speedlm](#)

**Examples**

```
y <- rnorm(100,1.5,1)
x <- round(matrix(rnorm(200), 100, 2), digits = 3)
colnames(x) <- c("s1","s2")
da <- data.frame(y, x)

m <- speedlm(y ~ s1 + s2,da)
summary(m)
```

# Index

## \*Topic **models**

- control, [2](#)
- speedglm, [5](#)
- speedglm-package, [2](#)
- speedlm, [9](#)
- summary.speedglm, [12](#)
- summary.speedlm, [14](#)

AIC.speedglm(summary.speedglm), [12](#)  
AIC.speedlm(summary.speedlm), [14](#)

bigglm, [6, 8](#)  
biglm, [10, 12](#)

chol, [4](#)  
coef.speedglm(summary.speedglm), [12](#)  
coef.speedlm(summary.speedlm), [14](#)  
control, [2, 6, 10, 11](#)  
cp, [6, 7, 10](#)  
cp(control), [2](#)  
crossprod, [4](#)

eigen, [4](#)

glm, [6, 8](#)

is.sparse, [6, 10](#)  
is.sparse(control), [2](#)

lm, [10, 12](#)  
logLik.speedglm(summary.speedglm), [12](#)  
logLik.speedlm(summary.speedlm), [14](#)

qr, [4](#)

shglm(speedglm), [5](#)  
solve, [6, 10](#)  
speedglm, [5, 12, 13](#)  
speedglm-package, [2](#)  
speedlm, [8, 9, 15](#)  
summary.speedglm, [12](#)

summary.speedlm, [12, 14](#)  
update.speedlm(speedlm), [9](#)  
vcov.speedglm(summary.speedglm), [12](#)  
vcov.speedlm(summary.speedlm), [14](#)