On the usage of the `geepack`

Søren Højsgaard and Ulrich Halekoh

`geepack` version 1.3.3 as of 2022-01-07

Contents

1 Introduction 1

2 Citing `geepack` 1

3 Simulating a dataset 2

4 Using the `waves` argument 3

5 Using a fixed correlation matrix and the `zcor` argument 5

6 When do GEE’s work best? 6

1 Introduction

This note contains a few extra examples. We illustrate the usage of a the `waves` argument and the `zcor` argument together with a fixed working correlation matrix for the `geeglm()` function.

2 Citing `geepack`

The primary reference for the `geepack` package is

> library(geepack)
> citation("geepack")

To cite geepack in publications use:

Generalized Estimating Equations Journal of Statistical Software, 15,
2, pp1--11

Structures Statistics in Medicine, 23, pp859--880.


To see these entries in BibTeX format, use 'print(<citation>,
bibtex=TRUE)', 'toBibtex(.)', or set
'options(citation.bibtex.max=999)'.

If you use geepack in your own work, please do cite the above reference.

3 Simulating a dataset

To illustrate the usage of the waves argument and the zcor argument together with
a fixed working correlation matrix for the geeglm() we simulate some data suitable
for a regression model.

> library(geepack)
> timeorder <- rep(1:5, 6)
> tvar <- timeorder + rnorm(length(timeorder))
> idvar <- rep(1:6, each=5)
> uuu <- rep(rnorm(6), each=5)
> yvar <- 1 + 2*tvar + uuu + rnorm(length(tvar))
> simdat <- data.frame(idvar, timeorder, tvar, yvar)
> head(simdat,12)

    idvar timeorder  tvar   yvar
1:     1        1 1.4378508 2.1167969
2:     1        2 0.9869071 3.8063269
3:     1        3 6.2177163 13.3844801
4:     1        4 4.1244457  7.7896086
5:     1        5 2.9206161  5.4945994
6:     2        1 0.8008056  0.4390851
7:     2        2 2.7318076  4.4079438
8:     2        3 2.5266557  5.9343648
9:     2        4 3.0076488  5.8567138
10:    2        5 5.1267370 10.3713445
11:    3        1 0.7846068 0.7542678
12:    3        2 1.9682030 1.5266326

Notice that clusters of data appear together in simdat and that observations are
ordered (according to timeorder) within clusters.

We can fit a model with an AR(1) error structure as
> mod1 <- geeglm(yvar ~ tvar, id=idvar, data=simdat, corstr="ar1")
> mod1

Call:
geeglm(formula = yvar ~ tvar, data = simdat, id = idvar, corstr = "ar1")

Coefficients:
(Intercept)   tvar
  0.1756438  1.9275874

Degrees of Freedom: 30 Total (i.e. Null); 28 Residual

Scale Link: identity
Estimated Scale Parameters: [1] 1.104

Correlation: Structure = ar1 Link = identity
Estimated Correlation Parameters:
  alpha
  0.3547194

Number of clusters: 6  Maximum cluster size: 5

This works because observations are ordered according to time within each subject in the dataset.

4 Using the waves argument

If observations were not ordered according to cluster and time within cluster we would get the wrong result:

> set.seed(123)
> ## library(doBy)
> simdatPerm <- simdat[sample(nrow(simdat)),]
> ## simdatPerm <- orderBy(~idvar, simdatPerm)
> simdatPerm <- simdatPerm[order(simdatPerm$idvar),]
> head(simdatPerm)

 idvar timeorder  tvar  yvar
 3  1     3  6.2177163 13.384480
 5  1     5  2.9206161  5.494599
 4  1     4  4.1244457  7.789609
 1  1     1  5.1267370 10.371344
 2  1     2  0.3679071  3.806327
 10 2     5  5.1267370 10.371344

Notice that in simdatPerm data is ordered according to subject but the time ordering within subject is random.

Fitting the model as before gives
```r
> mod2 <- geeglm(yvar~tvar, id=idvar, data=simdatPerm, corstr="ar1")
> mod2

Call: geeglm(formula = yvar ~ tvar, data = simdatPerm, id = idvar, corstr = "ar1")

Coefficients:
  (Intercept)   tvar
       0.1121615 1.9619006

Degrees of Freedom: 30 Total (i.e. Null); 28 Residual

Scale Link: identity
Estimated Scale Parameters: [1] 1.096969

Correlation: Structure = ar1 Link = identity
Estimated Correlation Parameters:
  alpha
       0.4569427

Number of clusters: 6  Maximum cluster size: 5

Likewise if clusters do not appear contiguously in data we also get the wrong result (the clusters are not recognized):

```r
> ## simdatPerm2 <- orderBy(~timeorder, data=simdat)
> simdatPerm2 <- simdat[order(simdat$timeorder),]
> geeglm(yvar~tvar, id=idvar, data=simdatPerm2, corstr="ar1")

Call: geeglm(formula = yvar ~ tvar, data = simdatPerm2, id = idvar, corstr = "ar1")

Coefficients:
  (Intercept)   tvar
       0.004709679 1.989074858

Degrees of Freedom: 30 Total (i.e. Null); 28 Residual

Scale Link: identity
Estimated Scale Parameters: [1] 1.09458

Correlation: Structure = ar1 Link = identity
Estimated Correlation Parameters:
  alpha
       0

Number of clusters: 30  Maximum cluster size: 1

To obtain the right result we must give the waves argument:
> wav <- simdatPerm$timeorder
> wav
[1] 3 5 4 1 2 6 4 3 2 1 5 4 1 3 2 4 3 5 2 1 2 4 5 3 1 3 2 5 4
> mod3 <- geeglm(yvar~tvar, id=idvar, data=simdatPerm, corstr="ar1", waves=wav)
> mod3

Call:  
geeglm(formula = yvar ~ tvar, data = simdatPerm, id = idvar,  
waves = wav, corstr = "ar1")

Coefficients:  
(Intercept)     tvar
0.1756438      1.9275874

Degrees of Freedom: 30 Total (i.e. Null); 28 Residual

Scale Link: identity
Estimated Scale Parameters: [1] 1.104

Correlation: Structure = ar1 Link = identity
Estimated Correlation Parameters:
alpha 0.3547194

Number of clusters: 6  Maximum cluster size: 5

5 Using a fixed correlation matrix and the zcor argument

Suppose we want to use a fixed working correlation matrix:

> cor.fixed <- matrix(c(1 , 0.5 , 0.25, 0.125, 0.125, 0.5 , 1 , 0.25, 0.125, 0.125, 0.25 , 0.25 , 1 , 0.5 , 0.125, 0.125, 0.125, 0.125, 0.125, 0.500 0.125 0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 1 ), 5, 5)
> cor.fixed
[1,] 1.0000000 0.5000000 0.2500000 0.1250000 0.1250000
[2,] 0.5000000 1.0000000 0.2500000 0.1250000 0.1250000
[3,] 0.2500000 0.2500000 1.0000000 0.5000000 0.1250000
[4,] 0.1250000 0.1250000 0.5000000 1.0000000 0.1250000
[5,] 0.1250000 0.1250000 0.1250000 0.1250000 1.0000000

Such a working correlation matrix has to be passed to geeglm() as a vector in the zcor argument. This vector can be created using the fixed2Zcor() function:

> zcor <- fixed2Zcor(cor.fixed, id=simdatPerm$idvar, waves=simdatPerm$timeorder)
> zcor
[1] 0.125 0.500 0.250 0.250 0.125 0.125 0.125 0.125 0.125 0.125 0.500 0.125 0.125
[13] 0.125 0.125 0.125 0.125 0.125 0.125 0.125 0.125 0.125 0.125 0.125 0.125 0.125
[25] 0.125 0.125 0.125 0.125 0.125 0.125 0.125 0.125 0.125 0.125 0.125 0.125 0.125
[37] 0.250 0.125 0.125 0.125 0.125 0.125 0.125 0.125 0.125 0.125 0.125 0.125 0.125
[49] 0.125 0.250 0.250 0.250 0.125 0.125 0.125 0.125 0.125 0.125 0.125 0.125 0.125

Notice that zcor contains correlations between measurements within the same cluster. Hence if a cluster contains only one observation, then there will be generated no entry in zcor for that cluster. Now we can fit the model with:
> mod4 <- geeglm(yvar ~ tvar, id=idvar, data=simdatPerm, corstr="fixed", zcor=zcor)
> mod4

Call:
geeglm(formula = yvar ~ tvar, data = simdatPerm, id = idvar,
      zcor = zcor, corstr = "fixed")

Coefficients:
(Intercept)      tvar
 0.1769065  1.9392192

Degrees of Freedom: 30 Total (i.e. Null); 28 Residual

Scale Link: identity
Estimated Scale Parameters: [1] 1.100996

Correlation: Structure = fixed   Link = identity
Estimated Correlation Parameters:
  alpha:1
    1

Number of clusters: 6   Maximum cluster size: 5

6 When do GEE’s work best?

GEEs work best when you have relatively many relatively small clusters in your data.