On the usage of the geepack

Søren Højsgaard and Ulrich Halekoh

gEEPack version 1.3.10 as of 2024-02-01

Contents

1 Introduction 1
   1.1 Citing geepack . . . . . . . . . . . . . . . . . . . . . . . . . 1
   1.2 When do GEE’s work best? . . . . . . . . . . . . . . . . . . . . 2

2 Simulating a dataset 2

3 Using the waves argument 3

4 Using a fixed correlation matrix and the zcor argument 5

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.

1.1 Citing geepack

The primary reference for the geepack package is


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

To cite geepack in publications use:


1.2 When do GEE’s work best?

1. GEEs work best when you have relatively many relatively small clusters of about equal size in your data.

2. If all your clusters are of size one you should not use GEEs; if all clusters are of size one a GEE corresponds to a generalized linear model.

3. If you only have very few clusters (and in the extreme case only one cluster) you are likely to encounter numerical difficulties.

NOTICE: Care must be taken with respect to the order in which the clusters appear in the dataset. See Section 3 for details.

2 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 data suitable for a regression model.

```r
> library(geepack)
> n_cluster <- 6
> n_time <- 5
> set.seed(1213)
> timeorder <- rep(1:n_time, n_cluster)
> tvar <- timeorder + rnorm(length(timeorder))
> idvar <- rep(1:n_cluster, each=n_time)
> uuu <- rep(rnorm(n_cluster), each=n_time) # A 'random intercept'
> 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 -0.1199294 0.7082247
2      1         2  3.0411260 7.1458370
3      1         3  0.5028488 3.3108124
4      1         4  1.7604888 3.8893380
5      1         5  6.1349016 14.2613890
6      2         1  1.2554963 3.4235073
7      2         2  0.6098986 3.9045993
8      2         3  2.8755870 6.0865324
9      2         4  3.9935618 9.7830956
10     2         5  6.0638870 14.0566776
```
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

```R
> 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
0.6314834  1.9908676
```

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

Scale Link: identity
Estimated Scale Parameters: [1] 1.390467

Correlation: Structure = ar1 Link = identity
Estimated Correlation Parameters:
```
 alpha
0.6521569
```

Number of clusters: 6 Maximum cluster size: 5

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

### 3 Using the waves argument

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

```R
> set.seed(123)
> simdatPerm <- simdat[sample(nrow(simdat)),]
> simdatPerm <- simdatPerm[order(simdatPerm$idvar),]
> head(simdatPerm)
```

```
idvar timeorder tvar   yvar
 3     1        3 0.5028488 3.3108124
 5     1        5 6.1349016 14.2613890
 4     1        4 1.7604888 3.8893380
 1     1        1 -0.1199294 0.7082247
 2     1        2 3.0411260 7.1458370
10     2        5 6.0638870 14.0556776
```

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

Fitting the model as before gives
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
  0.6802353  2.0048127

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

Scale Link: identity
Estimated Scale Parameters: [1] 1.38735

Correlation: Structure = ar1 Link = identity
Estimated Correlation Parameters:
  alpha
  0.6294383

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):

> simdatPerm2 <- simdat[order(simdat$timeorder),]
> head(simdatPerm2)

              idvar timeorder  tvar  yvar
 1          1          1  1 -0.1199294 0.7082247
 6          2          1  1  1.2554963 3.4235073
 11         3          1  1  1.8977871 5.6986661
 16         4          1  1  2.1254099 5.4517051
 21         5          1  1  0.5043048 0.7625234
 26         6          1  1  1.3327271 2.0536935

> geeglm(yvar~tvar, id=idvar, data=simdatPerm2, corstr="ar1")

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

Coefficients:
  (Intercept)  tvar
  0.6935408  1.9920382

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

Scale Link: identity
Estimated Scale Parameters: [1] 1.386165

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:

```r
> wav <- simdatPerm$timeorder
> wav

[1] 3 5 4 1 2 5 4 3 2 1 5 4 3 2 4 3 5 2 1 2 4 5 3 1 3 2 1 5 4
```

```r
> mod3 <- geeglm(yvar~tvar, id=idvar, data=simdatPerm, corstr="ar1", waves=wav)
> mod3
```

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

Coefficients:
(Intercept) tvar
```
  0.6314834  1.9908676
```

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

Scale Link: identity
Estimated Scale Parameters: [1] 1.390467

Correlation: Structure = ar1 Link = identity
Estimated Correlation Parameters:
alpha 0.6521569

Number of clusters: 6 Maximum cluster size: 5

4 Using a fixed correlation matrix and the zcor argument

Suppose we want to use a fixed working correlation matrix:

```r
> 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, 0.5 , 1 , 0.125,
+                          0.125, 0.125, 0.125, 0.125, 1 ), 5, 5)
> cor.fixed
```

```
[1,] 1.000 0.500 0.250 0.125 0.125
[2,] 0.500 1.000 0.250 0.125 0.125
```
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:

```r
> 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 0.500 0.125 0.125
[13] 0.125 0.125 0.500 0.125 0.125 0.250 0.250 0.500 0.125 0.125 0.125 0.125
[25] 0.250 0.125 0.125 0.500 0.125 0.125 0.250 0.500 0.125 0.125 0.125 0.125
[37] 0.250 0.125 0.125 0.500 0.125 0.125 0.250 0.500 0.125 0.125 0.125 0.125
[49] 0.125 0.250 0.250 0.250 0.125 0.500 0.500 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:

```r
> 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
  0.6716075  1.9854861

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

Scale Link: identity
Estimated Scale Parameters: [1] 1.388038

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

Number of clusters: 6  Maximum cluster size: 5