Bounded-memory GEE using gdManager/ff

VJ Carey

March 21, 2014

1 Introduction

This is a very simple approach to illustrating three principles of deployable statistical methodology. First, data are accessed flexibly, without the requirement that all records can be accessed simultaneously in main memory. Second, computations are isolated where possible so that they may be dispatched to slave cores in a multicore system. Third, effort is made to maximize reuse of existing numerical/statistical facilities in base R packages to program a GEE solver. This is not a fully general attempt at deployability.

2 Resources

We build an 'out-of-memory' clustered data set.

```
> library(geeni)
> library(nlme)
> # from help(groupedData)
>   Orth.new <-  # create a new copy of the groupedData object
+        groupedData( distance ~ age | Subject,
+                     data = as.data.frame( Orthodont ),
+                     FUN = mean,
+                     outer = ~ Sex,
+                     labels = list( x = "Age",
+                       y = "Distance from pituitary to pterygomaxillary fissure" ),
+                     units = list( x = "(yr)", y = "(mm)") )
> dim(Orth.new)

[1] 108   4

> library(ff)
> library(geeni)
> #targdir = system.file("ffdemo", package="geeni")
> #mantargdir = system.file("mgrs", package="geeni")
> #pref = paste(targdir, "gdm", sep="/")
> #fis = dir(pref, full=TRUE)
> #if (length(fis)>0) try(sapply(fis, file.remove))
> flatOrth = geeni:::gd2flat(gd=Orth.new, gcol=3, prefix="")# prefix=pref)
> flatOrth

gdManager instance
number of clusters =  27
size of data matrix:  108 x 3
excerpt:
  distance age Sex
1       26   8   1
2       25  10   1
3       29  12   1
```

The `flatOrth` object manages access to information on the orthodontistry dataset from nlme. The key task supported by the manager is retrieval of a specified cluster of observations using the getGrp method:

```
> getGrp(flatOrth,1)

  distance age Sex
1       26   8   1
2       25  10   1
3       29  12   1
4       31  14   1

> getGrp(flatOrth,4)

   distance age Sex
13     25.5   8   1
14     27.5  10   1
15     26.5  12   1
16     27.0  14   1
```

We now consider how to compute an updating step in a Newton-Raphson algorithm for solving the working independence generalized equation corresponding to the generalized linear model with components specified by an R `family` object.

For I observed clusters indexed by i, let y_i denote an $n_i \times 1$ response vector satisfying

$$E[y_i|x_i] = \mu_i(\beta) = g^{-1}(x_i\beta),$$

$$\mathrm{var}(y_i) = V(\mu_i)$$

where x_i is $n_i \times p$ matrix of covariates, and $g(\cdot)$ and $V(\cdot)$ are link and variance functions from the family of GLMs. We will eventually accommodate a working intracluster correlation model, but for now adopt working independence. We want to solve

$$\sum_i \frac{\partial \mu_i(\beta)}{\partial \beta}^t V^{-1}(\mu_i)[y_i - \mu_i(\beta)] = \sum_i D_i^t V_i^{-1} r_i = 0$$

for β by iterating

$$\hat{\beta}^{(s)} = \hat{\beta}^{(s-1)} + (\sum_i D_i^t V_i^{-1} D_i)^{-1} (\sum_i D_i^t V_i^{-1} r_i)$$

over $s = 1, \ldots$ until convergence.

```
> geeni:::getDep

function (x)
deparse(x@formula[[2]])
<environment: namespace:geeni>

> geeni:::getEta

function (gd, i, beta)
getX(gd, i) %*% beta
<environment: namespace:geeni>

> geeni:::getMu

function (gd, i, beta, family)
as.numeric(family()$linkinv(getEta(gd, i, beta)))
<environment: namespace:geeni>

> geeni:::getX

function (gd, i)
{
    dat = getGrp(gd, i)
    dep = getDep(gd)
    cbind(1, dat[, colnames(dat) != dep, drop = FALSE])
}
<environment: namespace:geeni>

> geeni:::getY
```

```
function (gd, i)
getGrp(gd, i)[, getDep(gd)]
<environment: namespace:geeni>

> geeni:::Di

function (gd, i, beta, family)
as.numeric(family()$mu.eta(getEta(gd, i, beta))) * getX(gd, i)
<environment: namespace:geeni>

> geeni:::Vinv.i

function (gd, i, beta, family)
diag(1/family()$variance(getMu(gd, i, beta, family)))
<environment: namespace:geeni>

> geeni:::ri

function (gd, i, beta, family)
getY(gd, i) - getMu(gd, i, beta, family)
<environment: namespace:geeni>
```

Here's code for a single updating step from an initial value of zero:

```
> beta = c(0,0,0)
> delb = function(gd, beta, family) {
+  DD = geeni:::Di(gd,1,beta,family)
+  val = t(DD) %*% geeni:::Vinv.i(gd,1,beta,family)
+  val1 = val %*% DD
+  val2 = val %*% geeni:::ri(gd,1,beta,family)
+  for  (i in 2:length(gd@discrim)) {
+     DD = geeni:::Di(gd, i, beta, family)
+     val = t(DD) %*% geeni:::Vinv.i(gd,i,beta,family)
+     val1 = val1 + val %*% DD
+     val2 = val2 + val %*% geeni:::ri(gd,i,beta,family)
+  }
+  solve(val1)%*%val2
+ }
> delb( flatOrth, beta, gaussian )

          [,1]
    20.0277357
age  0.6601852
Sex -2.3210227
```

For the Gaussian model the constituents seem to be correct.

```
> lm(distance~age+Sex,data=Orth.new)

Call:
lm(formula = distance ~ age + Sex, data = Orth.new)

Coefficients:
(Intercept)          age    SexFemale
    17.7067       0.6602      -2.3210
```

Any discrepancy in the intercept is attributable to different factor coding for Sex.
Now we factor this so that quantities for each cluster can be computed separately.

```
> geeni:::Gcomps

function (gd, i, beta, family)
{
    DD = Di(gd, i, beta, family)
    val = t(DD) %*% Vinv.i(gd, i, beta, family)
    val1 = val %*% DD
    val2 = val %*% ri(gd, i, beta, family)
    list(DtVDi = val1, DtVri = val2)
}
<environment: namespace:geeni>
```

We will use foreach and need an accumulator that will work with the list components defined above.

```
> geeni:::combi

function (x, y)
list(x[[1]] + y[[1]], x[[2]] + y[[2]])
<environment: namespace:geeni>
```

Here's a demonstration:

```
> library(foreach)
> library(doParallel)
> registerDoParallel(cores=2)  # for mac
> comps = foreach(i = 1:27, .combine=geeni:::combi) %dopar%
+     { geeni:::Gcomps(flatOrth,i,c(0,0,0),gaussian) }
> comps
```

```
[[1]]
          age  Sex
     108  1188  152
age 1188 13608 1672
Sex  152  1672  240

[[2]]
       [,1]
     2594.5
age 28896.0
Sex  3591.0

> del = solve(comps[[1]])%*%comps[[2]]
> beta = beta + del
> comps = foreach(i = 1:27, .combine= geeni:::combi) %dopar%
+     { geeni:::Gcomps(flatOrth,i,beta,gaussian) }
> comps

[[1]]
          age  Sex
     108  1188  152
age 1188 13608 1672
Sex  152  1672  240

[[2]]
            [,1]
    1.116973e-11
age 1.269314e-10
Sex 1.551825e-11
```

A generic solver is then

```
> geeni::pargee

function (gd, family, binit, maxit = 20, tol = 1e-06)
{
    beta = binit
    del = Inf
    curit = 1
    nclus = length(gd@discrim)
    while (max(abs(del)) > tol) {
        delcomp = foreach(i = 1:nclus, .combine = combi) %dopar%
            Gcomps(gd, i, beta, family)
        del = solve(delcomp[[1]]) %*% delcomp[[2]]
        beta = beta + del
        curit = curit + 1
        if (curit > maxit)
            stop(paste("maxit [", maxit, "] iterations exceeded"))
    }
    beta
}
<environment: namespace:geeni>
```

```
> pargee( flatOrth, gaussian, c(0,0,0) )
```

```
          [,1]
    20.0277357
age  0.6601852
Sex -2.3210227
```

To fix the factor coding discrepancy, if necessary:

```
> flatOrth@numdat[,"Sex"] = flatOrth@numdat[,"Sex"]-1  # overwrite allowed
```

We will remove the ff data.

```
> system("rm -rf .dat.ff")
> system("rm -rf .disc.ff")
```

```
> sessionInfo()
```

```
R version 3.0.3 Patched (2014-03-06 r65223)
Platform: x86_64-unknown-linux-gnu (64-bit)

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8        LC_COLLATE=C
 [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] parallel  tools     stats     graphics  grDevices utils     datasets
[8] methods   base

other attached packages:
```

```
[1] doParallel_1.0.8   iterators_1.0.6    IRanges_1.20.7     BiocGenerics_0.8.0
[5] nlme_3.1-115       geeni_0.0.8        foreach_1.4.1      ff_2.2-12
[9] bit_1.1-11

loaded via a namespace (and not attached):
[1] codetools_0.2-8 compiler_3.0.3  grid_3.0.3      lattice_0.20-27
[5] stats4_3.0.3
```