R code to illustrate robust & clustered standard error calculation

Daina Chiba
daina.chiba@gmail.com

September 1, 2009

1 Robust standard error

This section illustrates the calculation of robust standard error for GLM models. Examples are taken from logit model presented in Oneal & Russett (2005).

The log likelihood function for the jth observation is given as

$$\ln L_j = y_j \ln F(x_j\boldsymbol{\beta}) + (1 - y_j) \ln F(x_j\boldsymbol{\beta}) \tag{1}$$

where $y_j \in \{0, 1\}$ is the outcome variable and $F(\cdot)$ is the logit CDF.

Now, the formula for the robust estimator of variance is

$$\hat{V}(\hat{\boldsymbol{\beta}}) = \hat{\boldsymbol{V}} \left(\frac{n}{n-1} \sum_{j=1}^{n} \boldsymbol{u}_j' \boldsymbol{u}_j \right) \hat{\boldsymbol{V}} \tag{2}$$

where $\hat{\boldsymbol{V}} = (-\partial^2 \ln L / \partial \boldsymbol{\beta}^2)^{-1}$ (conventional estimator of variance) and $\boldsymbol{u}_j$ (a row vector) is the contribution from the jth observation to $\partial \ln L / \partial \beta$.

Tthe score, $\boldsymbol{u}_j$, or the first derivative of the log likelihood with respect to $x_j\beta$ is given by differentiating the log likelihood function

$$\boldsymbol{u}_j = y_j * \frac{f(x_j\boldsymbol{\beta})}{F(x_j\boldsymbol{\beta})} x_j + (1 - y_j) * (-\frac{f(x_j\boldsymbol{\beta})}{1 - F(x_j\boldsymbol{\beta})} x_j) \tag{3}$$

where $f(\cdot)$ is the logit PDF.

Now, let's estimate this step-by-step. First we read in the data set.

R Code

```
> rm(list = ls())
> options(scipen = 1)
> data <- read.table("http://dynaman.net/data/or2005.txt",
+     header = T, na.strings = "-99")
```

We define the CDF and PDF of the logit in R.

R Code

```
> model <- "logit"
> F <- function(x) {
+     1/(1 + exp(-x))
+ }
> f <- function(x) {
+     exp(x)/(1 + exp(x))^2
+ }
```

Now, estimate the logit model.

R Code

```
> fml <- as.formula(mzmid1 ~ smldem + lrgdem + smldep + allies +
+     lncaprat + dircont + lndstab + majpower + systsize + midpy +
+     midpy1 + midpy2 + midpy3)
> fit <- glm(fml, data = data, family = binomial(link = model))
> beta <- fit$coef
> vcov <- vcov(fit)
```

Define some variables for convenience.

R Code

```
> y <- data$mzmid1
> k <- length(beta)
> n <- nrow(data)
> xvars <- names(beta)
> xvars <- xvars[2:length(xvars)]
> xmat <- as.matrix(data[, xvars])
> xmat <- cbind(1, xmat)
> xb <- xmat %*% beta
```

As we obtain the scores as defined in (3), we vectorize it for speed.

R Code

```
> u <- ((y == 1) * f(xb)/F(xb) + (y == 0) * -f(xb)/(1 - F(xb)))[,
+     1] * xmat
```

Finally, we calculate the robust estimator of variance as defined in (2).

R Code

```
> rob.vcov <- vcov %*% ((n/(n - 1)) * t(u) %*% u) %*% vcov
```

Present the robust and conventional standard errors, along with coefficient estimates, in tables.

R Code

```
> rob.se <- sqrt(diag(rob.vcov))
> con.se <- sqrt(diag(vcov))
> rnd <- 3
> z <- beta/rob.se
> pval <- 2 * (1 - pnorm(abs(beta/rob.se)))
> rob.tbl <- cbind(beta = round(beta, rnd), se = round(rob.se,
+     rnd), z = round(z, 2), pval = round(pval, rnd))
> colnames(rob.tbl) <- c("coef", "s.e.", "z", "P>|z|")
> print("Results w/ Robust s.e. ")
```

```
[1] "Results w/ Robust s.e. "
```

R Code

```
> print(rob.tbl)
```

```
             coef  s.e.      z P>|z|
(Intercept) -0.489 0.239  -2.05 0.040
smldem      -0.069 0.005 -13.26 0.000
lrgdem       0.038 0.004   9.11 0.000
smldep      -0.323 0.079  -4.09 0.000
allies       0.075 0.068   1.11 0.267
lncaprat    -0.284 0.019 -15.04 0.000
dircont      1.127 0.088  12.86 0.000
lndstab     -0.289 0.032  -9.16 0.000
majpower     1.007 0.083  12.09 0.000
systsize    -0.495 0.024 -20.51 0.000
midpy       -0.376 0.015 -25.13 0.000
midpy1      -0.002 0.000 -13.91 0.000
midpy2       0.001 0.000  10.57 0.000
midpy3       0.000 0.000  -4.66 0.000
```

R Code

```
> z <- beta/con.se
> pval <- 2 * (1 - pnorm(abs(beta/con.se)))
> con.tbl <- cbind(beta = round(beta, rnd), se = round(con.se,
+     rnd), z = round(z, 2), pval = round(pval, rnd))
> colnames(con.tbl) <- c("coef", "s.e.", "z", "P>|z|")
> print("Results w/ conventional s.e.")
```

```
[1] "Results w/ conventional s.e."
```

R Code

```
> print(con.tbl)
```

```
             coef  s.e.      z P>|z|
(Intercept) -0.489 0.232  -2.11 0.035
smldem      -0.069 0.005 -13.13 0.000
lrgdem       0.038 0.004   9.20 0.000
```

```
smldep      -0.323 0.060  -5.40 0.000
allies       0.075 0.064   1.17 0.241
lncaprat    -0.284 0.018 -15.36 0.000
dircont      1.127 0.085  13.28 0.000
lndstab     -0.289 0.030  -9.74 0.000
majpower     1.007 0.080  12.58 0.000
systsize    -0.495 0.022 -22.11 0.000
midpy       -0.376 0.014 -26.91 0.000
midpy1      -0.002 0.000 -14.48 0.000
midpy2       0.001 0.000  10.90 0.000
midpy3       0.000 0.000  -4.77 0.000
```

Results are very close to each other.

2 Clustered standard error

Next, we calculate standard errors adjusted for clusters in dyad. When observations are independent only across unit (dyad) and interdependent within unit, the robust variance estimator is given as

$$\hat{V}(\hat{\boldsymbol{\beta}}) \quad = \quad \hat{\boldsymbol{V}} \left(\frac{m}{m-1} \sum_{k=1}^{m} \boldsymbol{u}_k^{(G')} \boldsymbol{u}_k^{(G)} \right) \hat{\boldsymbol{V}} \tag{4}$$

where $\boldsymbol{u}_k^{(G)}$ is the contribution from the kth group to $\partial \ln L / \partial \beta$. $\boldsymbol{u}_k^{(G)},\ k = 1, ..., m$ is calculated simply as

$$\boldsymbol{u}_k^{(G)} \quad = \quad \sum_{j \in G_k} \boldsymbol{u}_j. \tag{5}$$

To obtain this group-level score, we take summation of the scores, u_j, that we calculate in the previous section. As we have as many as 13277 groups (dyads), looping over groups is very time-consuming. We thus utilize `tapply` function, and loop over covariates.

R Code

```
> m <- dim(table(data$dyadid))
> u.clust <- matrix(NA, nrow = m, ncol = k)
> fc <- factor(data$dyadid)
> for (i in 1:k) {
+     u.clust[, i] <- tapply(u[, i], fc, sum)
+ }
```

Then, we obtain the variance estimator as defined in (4).

R Code

```
> cl.vcov <- vcov %*% ((m/(m - 1)) * t(u.clust) %*% (u.clust)) %*%
+     vcov
```

Present the results along with the conventional estimates.

R Code

```
> cl.se <- sqrt(diag(cl.vcov))
> z <- beta/cl.se
> pval <- 2 * (1 - pnorm(abs(beta/cl.se)))
> cl.tbl <- cbind(beta = round(beta, rnd), se = round(cl.se, rnd),
+     z = round(z, 2), pval = round(pval, rnd))
> colnames(cl.tbl) <- c("coef", "s.e.", "z", "P>|z|")
> print("Results w/ Robust s.e. clustered by Dyad")
```

```
[1] "Results w/ Robust s.e. clustered by Dyad"
```

R Code

```
> print(cl.tbl)
```

```
             coef  s.e.      z P>|z|
(Intercept) -0.489 0.406  -1.20 0.229
smldem      -0.069 0.008  -8.55 0.000
lrgdem       0.038 0.007   5.69 0.000
smldep      -0.323 0.090  -3.59 0.000
allies       0.075 0.101   0.75 0.456
```

```
lncaprat    -0.284 0.030  -9.32 0.000
dircont      1.127 0.152   7.40 0.000
lndstab     -0.289 0.053  -5.40 0.000
majpower     1.007 0.133   7.58 0.000
systsize    -0.495 0.039 -12.79 0.000
midpy       -0.376 0.020 -19.18 0.000
midpy1      -0.002 0.000 -11.51 0.000
midpy2       0.001 0.000   9.02 0.000
midpy3       0.000 0.000  -4.24 0.000
```

R Code

```
> print(con.tbl)
```

```
             coef  s.e.      z P>|z|
(Intercept) -0.489 0.232  -2.11 0.035
smldem      -0.069 0.005 -13.13 0.000
lrgdem       0.038 0.004   9.20 0.000
smldep      -0.323 0.060  -5.40 0.000
allies       0.075 0.064   1.17 0.241
lncaprat    -0.284 0.018 -15.36 0.000
dircont      1.127 0.085  13.28 0.000
lndstab     -0.289 0.030  -9.74 0.000
majpower     1.007 0.080  12.58 0.000
systsize    -0.495 0.022 -22.11 0.000
midpy       -0.376 0.014 -26.91 0.000
midpy1      -0.002 0.000 -14.48 0.000
midpy2       0.001 0.000  10.90 0.000
midpy3       0.000 0.000  -4.77 0.000
```