Homework 4 Solutions

Question 1

a)

$$\begin{aligned}
\mathbb{E}[e^2|X] &= \mathbb{E}[(Y - \mathbb{E}[Y|X])^2|X] \\
&= \mathbb{E}[(Y - \mathrm{P}(Y=1|X))^2|X] \\
&= \mathbb{E}[Y - 2Y\mathrm{P}(Y=1|X) + \mathrm{P}(Y=1|X)^2|X] \\
&= \mathrm{P}(Y=1|X) - 2\mathrm{P}(Y=1|X)^2 + \mathrm{P}(Y=1|X)^2 \\
&= \mathrm{P}(Y=1|X)(1 - \mathrm{P}(Y=1|X)) \\
&= X'\beta(1 - X'\beta)
\end{aligned}$$

where the first equality holds by the definition of e, the second equality holds because $\mathbb{E}[Y|X] = \mathrm{P}(Y=1|X)$ here, the third equality squares the term and holds because $Y^2 = Y$ when Y is binary, the fourth equality holds because $\mathbb{E}[Y|X] = \mathrm{P}(Y=1|X)$, the fifth equality cancels terms and factors, and the last equality holds because $\mathrm{P}(Y=1|X) = X'\beta$ here.

b)

$$\tilde{\beta} = \underset{b}{\operatorname{argmax}} \sum_{i=1}^{n} \Big(Y_i \log(X_i'b) + (1 - Y_i)\log(1 - X_i'b)) \Big)$$

The above expression is similar to what we have used befor for probit except for that $\Phi(X_i'b)$ is replaced with $X_i'b$. Taking the first order condition with respect to b, we have that

$$\begin{aligned}
0 &= \sum_{i=1}^{n} \left(\frac{Y_i}{X_i'\tilde{\beta}} X_i - \frac{(1 - Y_i)}{(1 - X_i'\tilde{\beta})} X_i \right) \\
&= \sum_{i=1}^{n} \frac{Y_i - X_i'\tilde{\beta}}{X_i'\tilde{\beta}(1 - X_i'\tilde{\beta})} X_i
\end{aligned}$$

$\tilde{\beta}$ is the solution to this equation. We are not able to derive an explicit equation for $\tilde{\beta}$ like we have for $\hat{\beta}$, but we can get the computer to solve this problem for us. [As a side-comment, $\tilde{\beta}$ may also run into computational problems as $\log(X_i'b)$ could be ≤ 0 for some values of X_i; this is related to the linear probability model not restricting predicted probabilities to be between 0 and 1.]

c) $\tilde{\beta}$ and $\hat{\beta}$ are clearly different. For both of them, we have that $\hat{\beta} \xrightarrow{p} \beta$ and that $\tilde{\beta} \xrightarrow{p} \beta$. But, more interestingly, notice that

$$\sqrt{n}(\hat{\beta} - \beta) \xrightarrow{d} N(0, \mathbf{V})$$

where

$$\begin{aligned}
\mathbf{V} &= \mathbb{E}[XX']^{-1}\mathbb{E}[XX'e^2]\mathbb{E}[XX']^{-1} \\
&= \mathbb{E}[XX']^{-1}\mathbb{E}\Big[XX'\mathbb{E}[e^2|X]\Big]\mathbb{E}[XX']^{-1} \\
&= \mathbb{E}[XX']^{-1}\mathbb{E}\Big[XX'(X'\beta)(1 - X'\beta)\Big]\mathbb{E}[XX']^{-1}
\end{aligned}$$

On the other hand, (using similar arguments to the ones in class for probit), we have that

$$\sqrt{n}(\tilde{\beta} - \beta) \xrightarrow{d} N(0, \mathbf{\Omega}^{-1})$$

where $\mathbf{\Omega} = \mathbb{E}\left[\left(\frac{1}{X'\beta(1-X'\beta)}\right) XX'\right]$.

One last interesting thing to notice is to recall the GLS estimator (this is the estimator that amounts to a weighted regression where the weights come from dividing by the variance of the the CEF error; the regression below is infeasible, but recall that feasible GLS that puts in $\hat{\beta}$ for β has the same asymptotic distribution)

$$\begin{aligned}
\hat{\beta}_{gls} &= \left(\frac{1}{n}\sum_{i=1}^{n} \frac{1}{X_i'\beta(1-X_i'\beta)} X_i X_i'\right)^{-1} \frac{1}{n}\sum_{i=1}^{n} \frac{1}{X_i'\beta(1-X_i'\beta)} X_i Y_i \\
\implies \sqrt{n}(\hat{\beta}_{gls} - \beta) &= \left(\frac{1}{n}\sum_{i=1}^{n} \frac{1}{X_i'\beta(1-X_i'\beta)} X_i X_i'\right)^{-1} \frac{1}{\sqrt{n}}\sum_{i=1}^{n} \frac{1}{X_i'\beta(1-X_i'\beta)} X_i e_i \\
&\xrightarrow{d} N(0, \mathbf{V}_{gls})
\end{aligned}$$

where

$$\begin{aligned}
\mathbf{V}_{gls} &= \mathbf{\Omega}^{-1}\mathbb{E}\left[\frac{1}{(X'\beta)^2(1-X'\beta)^2} XX'e^2\right]\mathbf{\Omega}^{-1} \\
&= \mathbf{\Omega}^{-1}\mathbb{E}\left[\frac{1}{(X'\beta)^2(1-X'\beta)^2} XX'\mathbb{E}[e^2|X]\right]\mathbf{\Omega}^{-1} \\
&= \mathbf{\Omega}^{-1}\mathbb{E}\left[\frac{1}{(X'\beta)(1-X'\beta)} XX'\right]\mathbf{\Omega}^{-1} \\
&= \mathbf{\Omega}^{-1}\mathbf{\Omega}\mathbf{\Omega}^{-1} = \mathbf{\Omega}^{-1}
\end{aligned}$$

where the first equality holds by our usual asymptotic arguments and the definition of $\mathbf{\Omega}$ from above, the second equality holds by the law of iterated expectations, the third equality holds by part (a), and the last equality holds by the definition of $\mathbf{\Omega}$ and canceling terms.

What we have shown is that, although both $\hat{\beta}$ and $\tilde{\beta}$ are consistent for β, they don't have the same asymptotic distribution. However, $\tilde{\beta}$ does have the same asymptotic distribution as the GLS estimator of β; this strongly suggesting that the maximum likelihood estimator and the GLS estimator are efficient while $\hat{\beta}$ is not. This is expected as, by construction, there is heteroskedasticity here, as pointed out in part (a).

Question 2

To start with recall that we are going to estimate the parameters in the probit model by solving the following optimization problem:

$$\hat{\beta} = \underset{b}{\text{argmax}} \frac{1}{n}\sum_{i=1}^{n} Y_i \log(\Phi(X_i'b)) + (1 - Y_i)\log(1 - \Phi(X_i'b))$$

and where it is also helpful to recall that we defined the score as the derivative of this objective function taken with respect to the parameters:

$$S_n(b) = \frac{1}{n}\sum_{i=1}^{n} \frac{(Y_i - \Phi(X_i'b))\phi(X_i'b)}{\Phi(X_i'b)(1 - \Phi(X_i'b))} X_i$$

```
library(haven)
data <- read_dta("cps09mar.dta")
data <- subset(data, female==0)
data$black <- 1*data$race==2
Y <- data$union
X <- as.matrix(data[,c("age","education","black","hisp")])
X <- cbind(1,X)
n <- nrow(data)

# log-likelihood as function of parameters
ll <- function(b, X, Y) {
  G <- pnorm(X%*%b)
  mean( Y*log(G) + (1-Y)*log(1-G) )
}

# score function
s <- function(b, X, Y) {
  G <- pnorm(X%*%b)
  g <- dnorm(X%*%b)

  # calculates mean across units (returning k-dim vector)
  apply(as.numeric(Y*(g/G))*X - as.numeric((1-Y)*(g/(1-G)))*X, 2, mean)
}

k <- ncol(X)
start_bet <- rep(0,k)
prob_est <- optim(start_bet, ll, gr=s,
                  X=X, Y=Y,
                  method="BFGS",
                  control=list(fnscale=-1))
bet <- prob_est$par
```

In order to calculate standard errors, recall that we showed in class that

$$\sqrt{n}(\hat{\beta} - \beta) \xrightarrow{d} N(0, \mathbf{\Omega}^{-1})$$

where $\mathbf{\Omega} = \mathbb{E}\left[\frac{\phi(X'\beta)^2}{\Phi(X'\beta)(1-\Phi(X'\beta))} XX'\right]$, and that we could estimate $\mathbf{\Omega}$ by

$$\hat{\mathbf{\Omega}} = \frac{1}{n}\sum_{i=1}^{n} \frac{\phi(X_i'\hat{\beta})^2}{\Phi(X_i'\hat{\beta})(1-\Phi(X_i'\hat{\beta}))} X_i X_i'$$

```
idx <- as.numeric(X%*%bet)
O1 <- ( dnorm(idx)^2 / ( pnorm(idx)*(1-pnorm(idx)) ) ) * X
Omeg <- t(O1)%*%X/n
Omeg_inv <- solve(Omeg)
se <- sqrt(diag(Omeg_inv))/sqrt(n)
round(cbind.data.frame(bet=bet, se=se, t=bet/se), 6)
```

```
##                 bet       se          t
##           -1.892941 0.106813 -17.721951
## age        0.007384 0.001416   5.215583
## education -0.028084 0.006212  -4.521237
## black     -0.063187 0.060269  -1.048417
## hisp      -0.289155 0.056130  -5.151499
```

Finally, let's compare these results to the ones that we get from R's `glm` command

```
# compare to results from R
R_probit <- glm(union ~ age + education + black + hisp, family=binomial(link=probit), data=data
summary(R_probit)
```

```
##
## Call:
## glm(formula = union ~ age + education + black + hisp, family = binomial(link = probit),
##     data = data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.4045  -0.2342  -0.2100  -0.1871   3.1500
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.953818   0.107176 -18.230  < 2e-16 ***
## age          0.007925   0.001419   5.584 2.35e-08 ***
## education   -0.025505   0.006221  -4.100 4.14e-05 ***
## blackTRUE   -0.054083   0.060004  -0.901    0.367
## hisp        -0.297745   0.056865  -5.236 1.64e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6282.0  on 29139  degrees of freedom
## Residual deviance: 6210.7  on 29135  degrees of freedom
## AIC: 6220.7
##
## Number of Fisher Scoring iterations: 6
```

These are slightly different from each other. I checked the documentation for `glm` and it uses an "iteratively reweighted least squares" estimation procedure; this is different from the optimization procedure that I used and explains the difference between the estimates.

Average Partial Effects:

To start with, let's compute estimates of average partial effects. Given what we have already done, this is fairly easy.

```r
# compute average partial effects
pe <- dnorm(X%*%bet) %*% t(bet)
ape <- apply(pe, 2, mean)
```

Next, let's derive the limiting distribution of the the average partial effects and use this to compute standard errors (this is **Option 1** in the homework).

Starting from the hint in the problem

$$\begin{aligned}
\sqrt{n}(\widehat{APE} - APE) &= \sqrt{n}\left(\frac{1}{n}\sum_{i=1}^{n}\phi(X_i'\hat{\beta})\hat{\beta} - \mathbb{E}[\phi(X'\beta)\beta]\right) \\
&= \sqrt{n}\left(\frac{1}{n}\sum_{i=1}^{n}\phi(X_i'\hat{\beta})\hat{\beta} - \frac{1}{n}\sum_{i=1}^{n}\phi(X_i'\hat{\beta})\beta\right) \qquad \text{(A)} \\
&+ \sqrt{n}\left(\frac{1}{n}\sum_{i=1}^{n}\phi(X_i'\hat{\beta})\beta - \frac{1}{n}\sum_{i=1}^{n}\phi(X_i'\beta)\beta\right) \qquad \text{(B)} \\
&+ \sqrt{n}\left(\frac{1}{n}\sum_{i=1}^{n}\phi(X_i'\beta)\beta - \mathbb{E}[\phi(X'\beta)\beta]\right) \qquad \text{(C)}
\end{aligned}$$

It is helpful to recall from class that we defined

$$\begin{aligned}
\mathbf{Q} &= -\mathbb{E}\left[\frac{\phi(X'\beta)^2}{\Phi(X'\beta)(1-\Phi(X'\beta))}XX'\right] \\
\psi(Y,X,b) &= -\frac{(Y-\Phi(X'b))\phi(X'b)}{\Phi(X'b)(1-\Phi(X'b))}X
\end{aligned}$$

and we showed that

$$\sqrt{n}(\hat{\beta}-\beta) = -\mathbf{Q}^{-1}\frac{1}{\sqrt{n}}\sum_{i=1}^{n}\psi(Y_i,X_i,\beta) + o_p(1)$$

which was the intermediate step before we applied the CLT.

Now, let's consider each term in turn in the expression for $\sqrt{n}(\widehat{APE} - APE)$, starting with the first one.

$$\begin{aligned}
(A) &= \frac{1}{n}\sum_{i=1}^{n}\phi(X_i'\hat{\beta})\sqrt{n}(\hat{\beta}-\beta) \\
&= \mathbb{E}[\phi(X'\beta)]\sqrt{n}(\hat{\beta}-\beta) + o_p(1) \\
&= \mathbb{E}[\phi(X'\beta)]\mathbf{Q}^{-1}\frac{1}{\sqrt{n}}\sum_{i=1}^{n}\psi(Y_i,X_i,\beta) + o_p(1) \\
&= \frac{1}{\sqrt{n}}\sum_{i=1}^{n}\mathbb{E}[\phi(X'\beta)]\mathbf{Q}^{-1}\psi(Y_i,X_i,\beta) + o_p(1) \\
&:= \frac{1}{\sqrt{n}}\sum_{i=1}^{n}A_i + o_p(1)
\end{aligned}$$

where the second equality holds by the law of large numbers and CMT, the third line holds by what we showed in class for $\sqrt{n}(\hat{\beta}-\beta)$, the fourth line just rearranges in a way that will be convenient below, and the fifth equality just introduces a more concise notation.

Next, consider the second term in the hint (this is hardest term to deal with). Notice that we can write

$$\begin{aligned}
(B) &= \beta \frac{1}{\sqrt{n}} \sum_{i=1}^{n} \phi(X_i'\hat{\beta}) - \phi(X_i'\beta) \\
&= \beta \frac{1}{n} \sum_{i=1}^{n} \phi'(X_i'\beta) X_i' \sqrt{n}(\hat{\beta} - \beta) + o_p(1) \\
&= -\beta \frac{1}{n} \sum_{i=1}^{n} X_i'\beta\phi(X_i'\beta) X_i' \sqrt{n}(\hat{\beta} - \beta) + o_p(1) \\
&= -\beta \mathbb{E}[X'\beta\phi(X'\beta)X']\sqrt{n}(\hat{\beta} - \beta) + o_p(1) \\
&= \beta \mathbb{E}[X'\beta\phi(X'\beta)X']\mathbf{Q}^{-1} \frac{1}{\sqrt{n}} \sum_{i=1}^{n} \psi(Y_i, X_i, \beta) + o_p(1) \\
&= \frac{1}{\sqrt{n}} \sum_{i=1}^{n} \beta \mathbb{E}[X'\beta\phi(X'\beta)X']\mathbf{Q}^{-1} \psi(Y_i, X_i, \beta) + o_p(1) \\
&:= \frac{1}{\sqrt{n}} \sum_{i=1}^{n} B_i + o_p(1)
\end{aligned}$$

where the first equality holds just by re-arranging, the second equality holds using delta method / mean value theorem type of argument, the third equality holds because $\phi'(z) = -z\phi(z)$ (this is a property of a standard normal distribution), the fourth equality holds by the law of large numbers and CMT, the fifth equality holds by what we showed in class for the asymptotically linear representation of $\sqrt{n}(\hat{\beta} - \beta)$, the sixth equality just re-arranges by pushing inside the sum the expectation terms, and the last line defines B_i.

Finally, the third term is immediately equal to

$$\begin{aligned}
(C) &= \frac{1}{\sqrt{n}} \sum_{i=1}^{\beta} (\phi(X_i'\beta) - \mathbb{E}[\phi(X'\beta)]) \\
&:= \frac{1}{\sqrt{n}} \sum_{i=1}^{n} C_i
\end{aligned}$$

Thus, we can write

$$\sqrt{n}(\widehat{APE} - APE) = \frac{1}{\sqrt{n}} \sum_{i=1}^{n} (A_i + B_i + C_i) + o_p(1)$$

If you plug in the expressions for A_i, B_i, C_i, this expression will look very complicated, but it is mean 0, and we can apply the CLT to it. In fact, it immediately follows that

$$\sqrt{n}(\widehat{APE} - APE) \xrightarrow{d} N(0, \mathbf{V})$$

where

$$\mathbf{V} = \mathbb{E}[(A + B + C)(A + B + C)']$$

Estimating $\mathbf{V}$ just involves plugging in sample quantities for population quantities — again: these expressions will be long, but if you do it carefully, everything should work. This is what we will do next.

```
# compute standard errors
idx <- as.numeric(X%*%bet) #nx1
Q <- -Omeg # kxk
G <- pnorm(X%*%bet) # nx1
g <- dnorm(X%*%bet) # nx1
# psi is nxk matrix
psi <- -as.numeric( Y*(g/G))*X - as.numeric((1-Y)*(g/(1-G)))*X

# compute A
a1 <- mean(g)
A <- a1*psi%*%solve(Q)

# compute B
b1 <- t(apply(as.numeric(idx * g) * X, 2, mean))
B <- t(as.matrix(bet)%*%b1%*%solve(Q)%*%t(psi))

# compute C
C <- as.numeric(g - mean(g))%*%t(bet)

# compute variance
inf_func <- A + B + C
# estimate variance
ape_V <- t(inf_func)%*%inf_func/n
ape_se <- sqrt(diag(ape_V))/sqrt(n)
round(cbind.data.frame(ape, ape_se, t=(ape/ape_se)),4)
```

```
##                ape ape_se        t
##           -0.1009 0.0065 -15.5874
## age        0.0004 0.0001   5.4706
## education -0.0015 0.0003  -4.9468
## black     -0.0034 0.0033  -1.0344
## hisp      -0.0154 0.0032  -4.8905
```

Let's compare these to what you get if you use R to compute average partial effects.

```
library(margins)
rversion_ape <- margins(R_probit)
summary(rversion_ape)
```

```
##     factor     AME     SE       z      p   lower   upper
##        age  0.0004 0.0001  5.5108 0.0000  0.0003  0.0006
##      black -0.0027 0.0029 -0.9430 0.3457 -0.0085  0.0030
##  education -0.0014 0.0003 -4.0706 0.0000 -0.0020 -0.0007
##       hisp -0.0158 0.0031 -5.1742 0.0000 -0.0218 -0.0098
```

As before, we're getting slightly different results. This is expected though; recall, that our original probit estimates were slightly different from the ones coming from R which would suggest that we'd expect slightly different APEs. That said, these are reasonable close and suggest that we do not have a coding error or anything like that.

Option 2: Bootstrap

This is quite similar to what we have done for the bootstrap before. The code below uses parallel processing to speed up computation using the `pbapply` package. The bootstrap is an example of what's sometimes called an "embarrassingly parallel" problem – this is kind of a strange name, but it just means that it's an obvious place to use parallel processing. The reason is that each bootstrap iteration is fully independent of other bootstrap iterations, so you can run lots of these at the same time and then compute standard errors (or whatever you want) after you have carried out all of the bootstrap iterations.

```
# finally, compute standard errors using the bootstrap
biters <- 100
library(pbapply) # for computing in parallel
boot_res <- pblapply(1:biters, function(b) {

  # draw new data with replacement
  boot_rows <- sample(1:n, size=n, replace=TRUE)
  boot_data <- data[boot_rows,]
  boot.Y <- boot_data$union
  boot.X <- as.matrix(data[,c("age","education","black","hisp")])
  boot.X <- cbind(1,boot.X)

  # estimate probit using new data
  boot_est <- optim(start_bet, ll, gr=s,
                    X=boot.X,
                    Y=boot.Y,
                    method="BFGS",
                    control=list(fnscale=-1))
  boot_bet <- boot_est$par

  # compute average partial effects
  boot_pe <- dnorm(boot.X%*%boot_bet) %*% t(boot_bet)
  boot_ape <- apply(boot_pe, 2, mean)

  # return results
  boot_ape
}, cl=2)

# run bootstrap
boot_res <- do.call("rbind", boot_res)

# compute bootstrap standard errors
boot_se <- apply(boot_res, 2, sd)

# compare to earlier standard errors
round(cbind.data.frame(ape=ape, ape_se=ape_se, boot_se=boot_se),4)
```

```
##               ape ape_se boot_se
##           -0.1009 0.0065  0.0064
```

```
## age        0.0004 0.0001  0.0001
## education -0.0015 0.0003  0.0003
## black     -0.0034 0.0033  0.0032
## hisp      -0.0154 0.0032  0.0025
```

These standard errors are very similar to the ones we calculated using asymptotic theory. Also, notice that, for me, it takes about a minute to compute the bootstrap standard errors, but only a second or two to compute the asymptotic standard errors. However, it only took 5 or 10 minutes for me to write the bootstrap code while it took me close to two hours to figure out the limiting distribution of the average partial effects and write the code for them (these are also probably more prone to making mistakes here because the arguments are more complicated).