PROBIT SIMULATION

The R function `glm` attempts maximum likelihood estimation for many models (the class of *generalized linear models*), which includes the probit model. (Why only "attemps"?)

Here we demonstrate, via simulation, that the estimates are unbiased if the latent variable is independent of the covariates, while it can be biased otherwise.

Our model is that

$$Y_i^\star = bX_i + cW_i + V_i,$$

$$Y_i = \begin{cases} 1 & \text{if } Y_i > 0 \\ 0 & \text{otherwise.} \end{cases}$$

We assume that (X_i, W_i, V_i) are multivariate Normal with covariance matrix

$$\boldsymbol{\Sigma} = \begin{bmatrix} 1 & 0.3 & 0 \\ 0.3 & 1 & \rho \\ 0 & \rho & 1 \end{bmatrix}.$$

We first assume that $\rho = 0$, so that our usual assumptions are met: (X_i, W_i) are independet of V_i.

Here $i = 1, 2, \ldots, 632$. We assume (X_i, W_i, V_i) are indepent as triples across i.

How do we simulate (X, W, V)? We use that $(X, W, V) = \boldsymbol{\Sigma}^{1/2}(Z_1, Z_2, Z_3)'$, where (Z_1, Z_2, Z_3) are i.i.d. (Why?).

Since $\boldsymbol{\Sigma}$ is symmetric, it has a spectral decomposition $\boldsymbol{\Sigma} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}'$, whence its square-root can be found via $\boldsymbol{\Sigma}^{1/2} = \boldsymbol{U}\sqrt{\boldsymbol{\Lambda}}\boldsymbol{U}'$:

```
> Sig = matrix(c(1,0.3,0,0.3,1,0,0,0,1),ncol=3,byrow=T)
> SigSD = eigen(Sig)
> SigSR = SigSD$vectors%*%diag(sqrt(SigSD$values))%*%t(SigSD$vectors)
```

To simulate 653 copies of (X, W, V), we generate a 3×653 matrix of independent standard Normals, $\boldsymbol{Z}$, and form the product $\boldsymbol{\Sigma}^{1/2}\boldsymbol{Z}$. Since we want the rows to be the 653 data triples (X_i, W_i, V_i), and the columns to be $\boldsymbol{X} = (X_1, \ldots, X_{653})'$, $\boldsymbol{Y}$, and $\boldsymbol{Z}$, we take the transpose.

```
> xwv = t(SigSR%*%matrix(rnorm(I(653*3)),ncol=653))
```

We can generate $Y^\star$ and Y, for a specific choice of b and c, and then estimate b and c via maximum likelihood:

```
>  ystar = .2*xwv[,1]+.5*xwv[,2]+xwv[,3]
>  y = as.numeric(ystar>0)
>  fit = glm(y~xwv[,1]+xwv[,2]-1,family = binomial(link="probit"))
>  bc=coef(fit)
>  bc
```

```
 xwv[, 1]  xwv[, 2]
0.1725073 0.5485419
```

Doing this a single time is not interesting, because we are doing this exercise to determine the properties of the estimating procedure, not the values of b and c (which we know!!)

So we perform this 1000 times, collecting our estimates $(\hat{c}_i, \hat{c}_i)_{i=1}^{1000}$. We can thus estimate the bias via

$$\bar{\hat{c}} - c = \frac{1}{1000} \sum \hat{c}_i - c.$$

We also prove the sample standard deviation of the $\hat{c}_i$'s, divided by $\sqrt{1000}$. Why do we divide by $\sqrt{1000}$?

```
> #Sig = matrix(c(1,0.3,0,0.3,1,0.1,0,0.1,1),ncol=3,byrow=T)
> bc = matrix(rep(1,2000),nrow=1000)
> for(i in 1:1000){
+   xwv = t(SigSR%*%matrix(rnorm(1500),ncol=500))
+   y = .2*xwv[,1]+.5*xwv[,2]+xwv[,3]
+   y = as.numeric(y>0)
+   fit = glm(y~xwv[,1]+xwv[,2]-1,family = binomial(link="probit"))
+   bc[i,]=coef(fit)
+ }
> mean(bc[,2])-0.5; sd(bc[,2])/sqrt(1000)
[1] 0.002879158
[1] 0.002241134
```

How can we get a confidence interval for the bias $\mathbb{E}_{0.2,0.5}[\hat{c}] - 0.5$?

Now, estimate the bias when $\rho = 0.4$. When $\rho = 0.1$?