

Econ 673
Problem Set #1
Answer Sheets

The attached code provides the solutions to both of the problems.

1) Consider the linear model:

$$y_i^* = \beta_0 + \beta_1 X_{1i} + \varepsilon_i; \quad \varepsilon_i \sim N(0, \sigma^2) \quad (1)$$

Unfortunately, y_i^* is not observed. Instead, you only observe a censored version, y_i , where

$$y_i = \begin{cases} 0 & y_i^* < 0 \\ y_i^* & y_i^* \geq 0 \end{cases} \quad (2)$$

You are interested in evaluating (via Monte Carlo analysis) the properties of the standard OLS estimator of β when using the censored data (i.e., y_i) rather than the uncensored values (i.e., y_i^*)

a) Assume the following parameter values for your model:

- $\beta_0 = 0.5$
- $\beta_1 = 1$
- $\sigma = 2$

b) Generate N=200 draws for y_i^* assuming that $X_i \sim U[0,1]$ and using equation (1)

c) Generate corresponding values for y_i using equation (2).

d) Using Matlab (or GAUSS) and no canned routines!, obtain the OLS estimates of the parameters of the mis-specified model:

$$y_i = \beta_0 + \beta_1 X_i + \varepsilon_i \quad (3)$$

Your results should include:

- i) OLS estimates of β_0 and β_1 , as well as an estimate of σ .**
- ii) An estimate of the asymptotic variance-covariance matrix for the OLS estimates**

The following Matlab functions and commands will be useful:

inv, diag,

The following results are obtained in terms of the coefficient estimates in a single run (parts a through e of the problem):

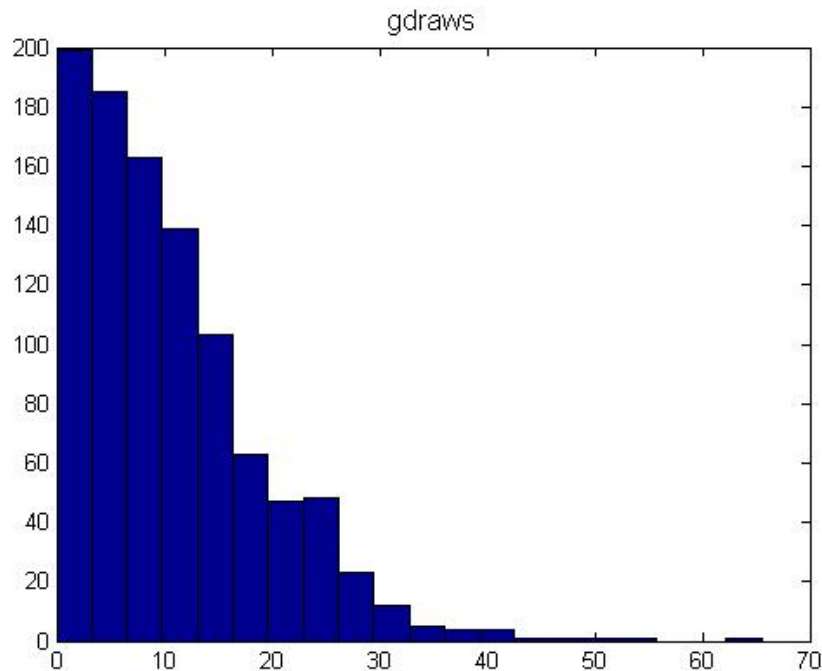
```
Estimates of beta0, beta1 and sigma
ans =
    1.2187    0.3376    1.4538
Estimate of asymptotic var-cov matrix
avarcov =
    0.0294   -0.0451
   -0.0451    0.0919
```

Note that these results are consistent with the traditional findings in censored regression cases, with beta1 shrinking towards zero.

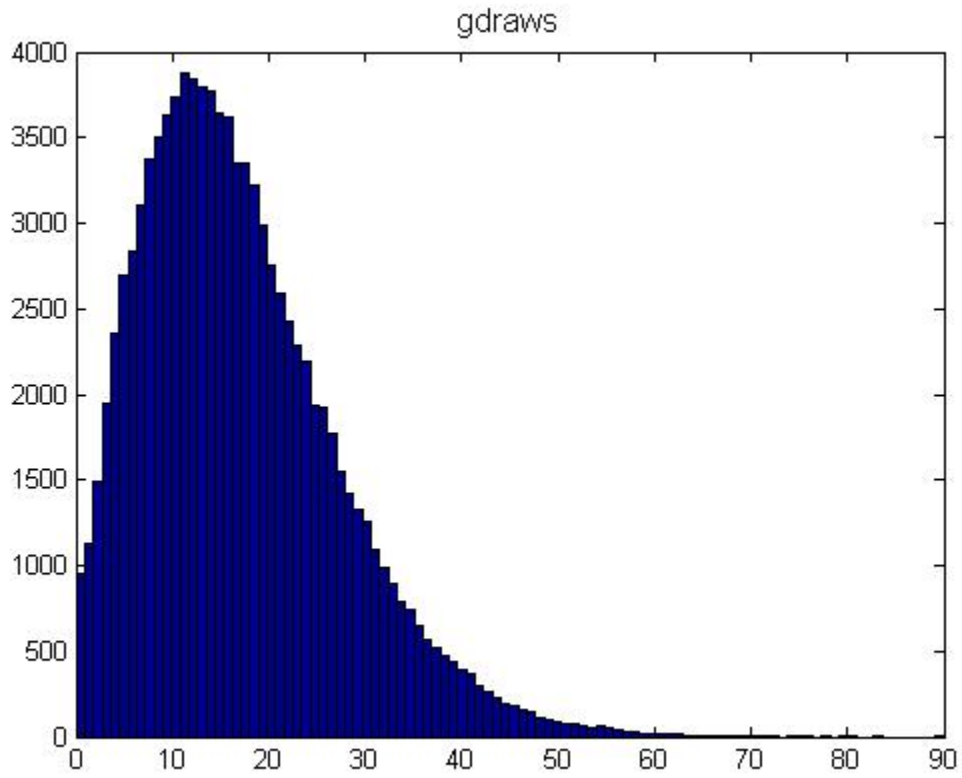
- e) **A colleague would like you to construct a 90% confidence interval for the variable $g(\beta) = (\beta_0 + 5\beta_1)^2$. Use simulation draws from the asymptotic distribution of the estimated parameters to provide them with the desired confidence interval.**

```
Estimates of lower and upper bounds of 90% confidence
interval
ans =
    0.4424    27.0533
```

I also graphed the draws for the function $g(\beta)$ in the following figure:



A somewhat clear picture emerges if we use more draws (100000) and a finer histogram:

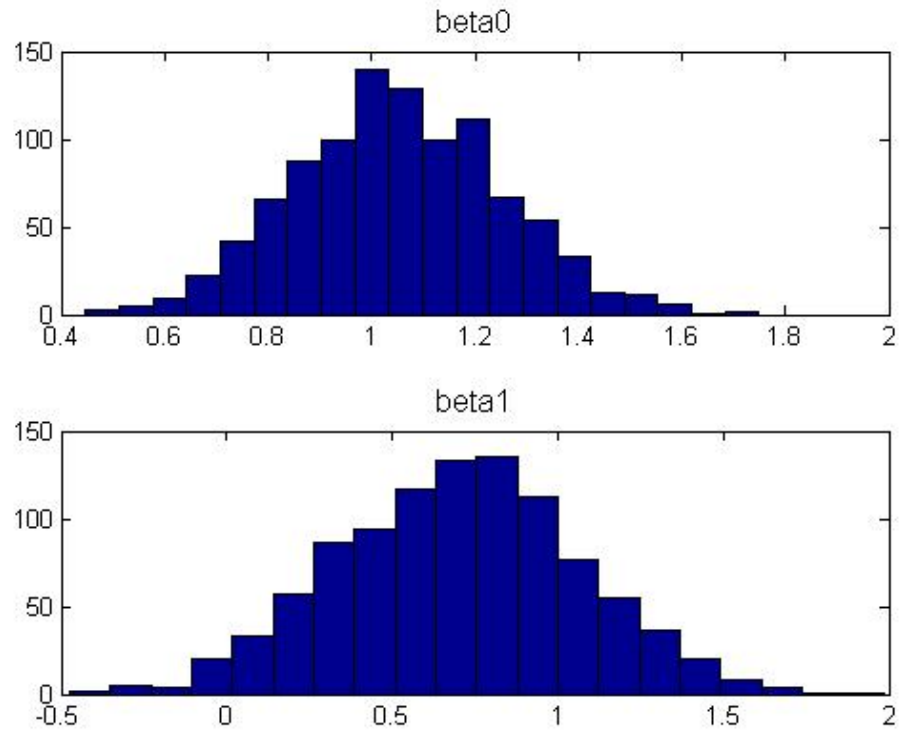


- f) Repeat steps (b) through (d) above a total of $R=1000$ times, collecting information about the OLS of β_0 and β_1 . Specifically, provide
- Summary statistics for the OLS estimates of β_0 and β_1 , including mean, standard deviations, minimums and maximums (using Matlab functions `mean`, `std`, `min` and `max`).

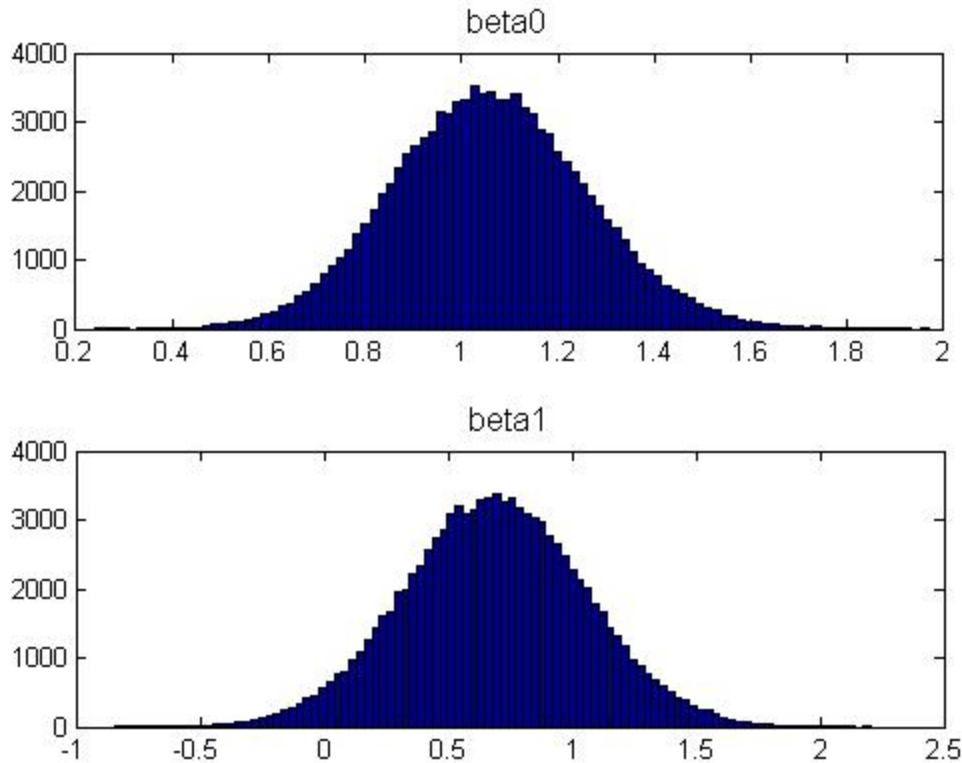
The resulting summary statistics are as follows:

```
Mean, std, min and max for beta0
ans =
    1.0523    0.2018    0.4477    1.7489
Mean, std, min and max for beta1
ans =
    0.7013    0.3689   -0.4734    1.9841
```

- A histogram of the estimates of β_0 and β_1 (using the Matlab functions: `hist`, `figure`, and `subplot`).



A somewhat clear picture emerges if we use more draws (100000) and a finer histogram:



You may find it convenient to use a loop in conducting the above Monte Carlo exercise, storing the OLS estimates of β_0 and β_1 after each loop $r=1,2,\dots,R$. For example:

```
R          = 1000;
theta     = zeros(nobs,2);
for       r = 1:R;
    ...your code to obtain OLS estimates of  $\beta_0$  and  $\beta_1$  and iteration  $r$ ,
        labeled thetahat
    theta(r,:) = thetahat;
end;
```

What do your results tell you about the direction of the bias in estimating β_0 and β_1 using the censored data?

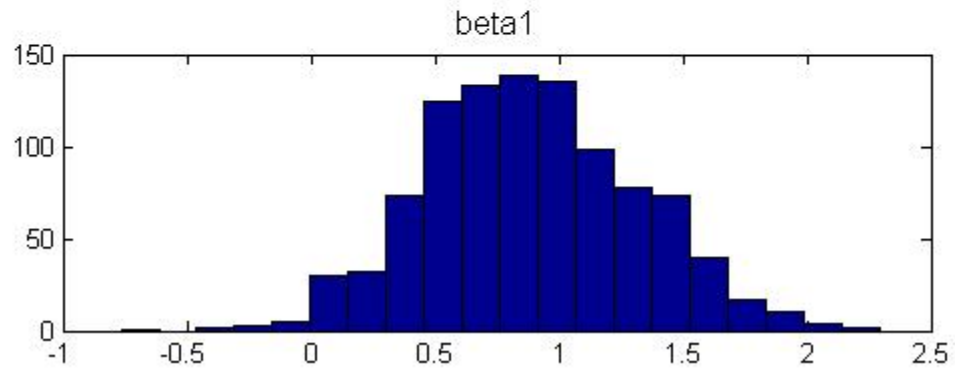
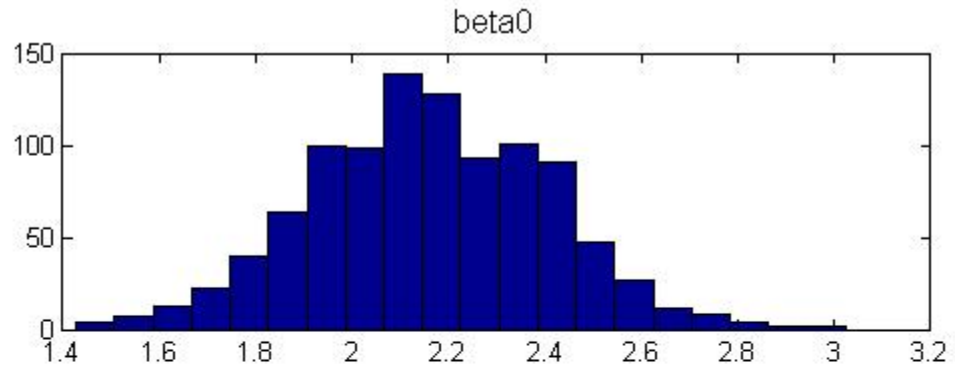
Clearly, we again see the bias in the OLS estimator towards zero.

g) Repeat step (f) using $\beta_0 = 2$. How have your results changed (and why)?

The new results are:

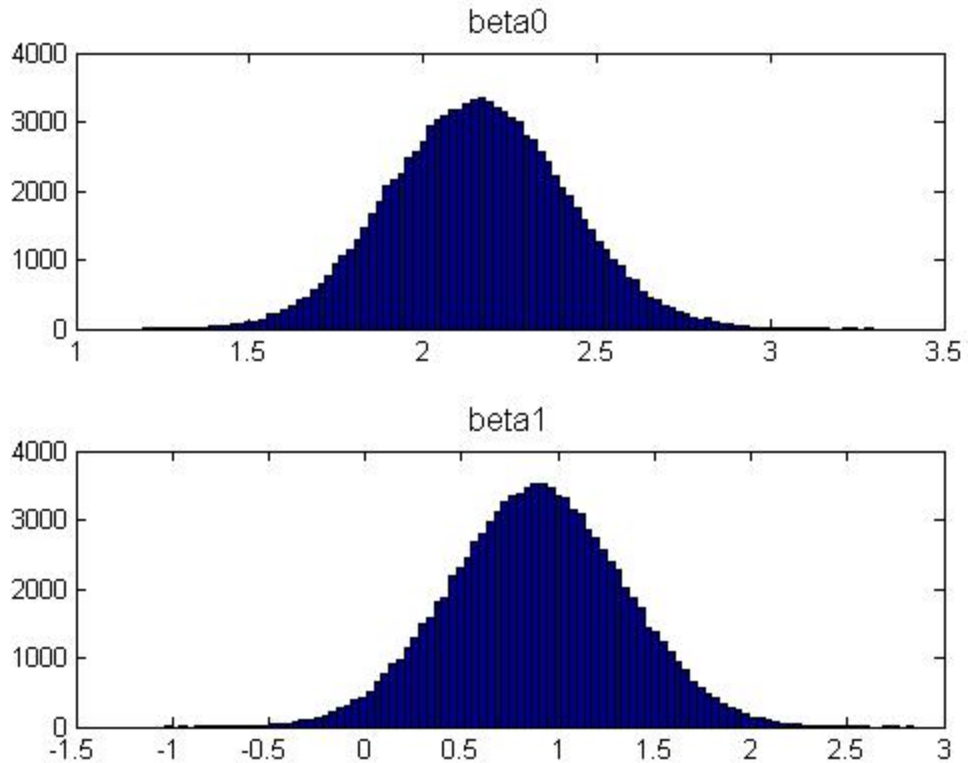
```
Mean, std, min and max for beta0
```

```
ans =  
2.1596    0.2479    1.4280    3.0221  
Mean, std, min and max for beta1  
ans =  
0.8913    0.4309   -0.7656    2.2913
```



The bias is smaller in this second case because there is generally less censoring occurring (i.e., few of the y_i^* 's are negative).

A somewhat clear picture emerges if we use more draws (100000) and a finer histogram:



2) Using the techniques discussed in class, generate $R=500$ draws from the following

a) $V_1 \sim TN_{(5,60)}(0, 25)$, where $TN_{(a,b)}(\mu, \sigma^2)$ denotes a truncated normal distribution, truncated below at a and above at b .

b) $V_2 = \begin{cases} 1 & \text{with probability 0.20} \\ 2 & \text{with probability 0.40} \\ 3 & \text{with probability 0.06} \\ 4 & \text{with probability 0.30} \\ 5 & \text{with probability 0.04} \end{cases}$

c) V_3 drawn from an extreme value distribution truncated from below at 0.5 and from above at 1.5.

d) $V_4 \sim T(0.5, 1.5)$, where $T(a, b)$ denotes a triangular distribution with mean a , with pdf:

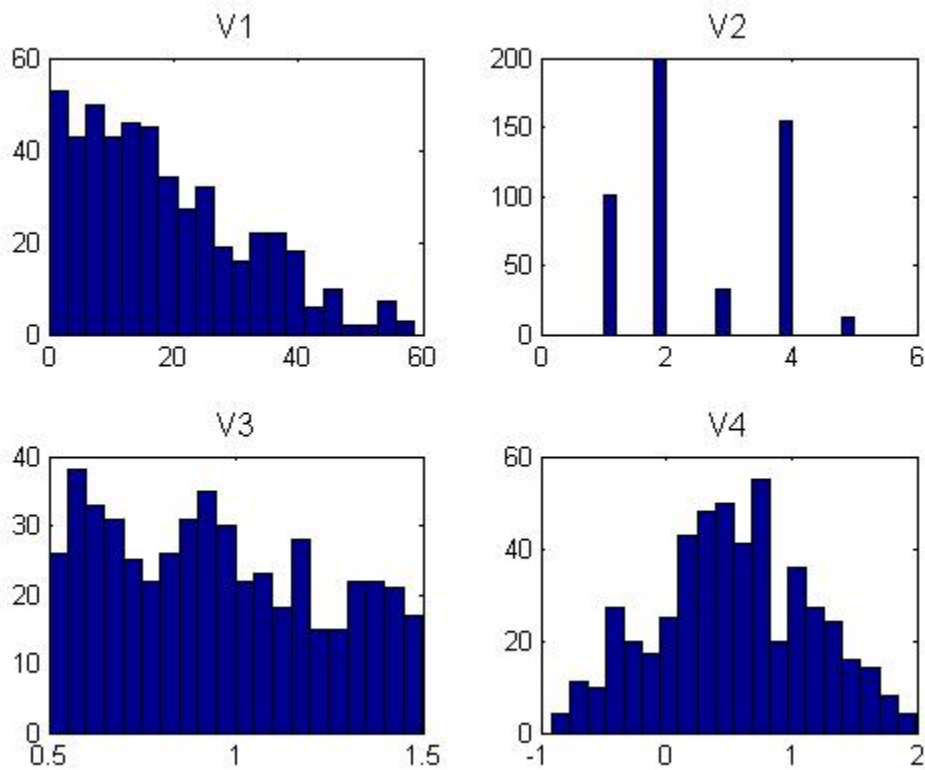
$$f(x) = \begin{cases} 0 & x \leq a-b \\ \frac{x+b-a}{b^2} & a-b < x \leq a \\ \frac{a+b-x}{b^2} & a < x \leq a+b \\ 0 & x > a+b \end{cases}$$

For V_4 you must use the acceptance rejection method to obtain the draws of interest.

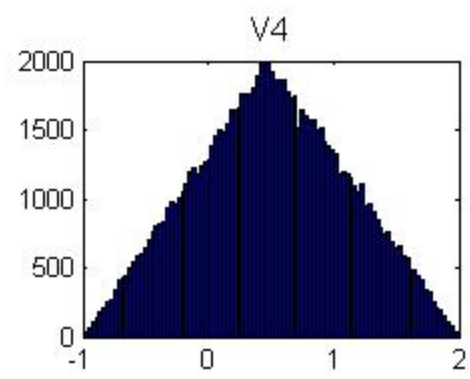
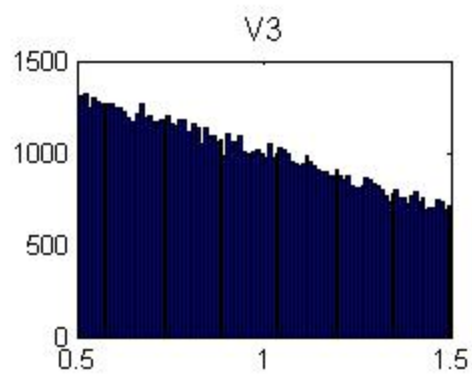
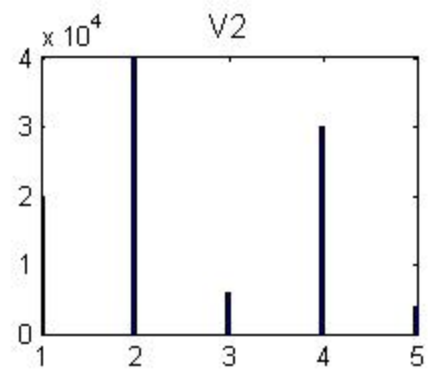
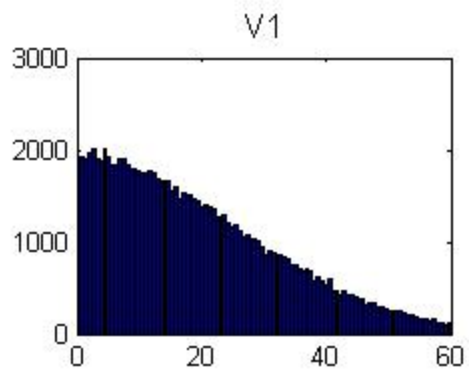
You can make use of the uniform random number generator `rand`. However, do not use any other random number generators. You may also make use of the Matlab function that computes the inverse of the normal cdf, `norminv`.

For each variable, provide a histogram of the sequence of draws using the Matlab functions `hist`, `figure`, and `subplot`.

The resulting histograms are provided below:



A somewhat clear picture emerges if we use more draws (100000) and a finer histogram:



MATLAB code:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%      Econ 673 Problem Set #1 Fall 2008
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear;
clc;
R      = 1000;
R2     = 500;
hspace = 20;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%      Problem 1
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%      Specify the model parameters
%
beta0  = 0.5;
beta1  = 1;
sigma  = 2;
theta  = [beta0 beta1]';
nobs   = 200;
%
%      Generate ystar
%
eps    = sigma*randn(nobs,1);
X      = [ones(nobs,1) rand(nobs,1)];
ystar  = X*theta + eps;
%
%      Construct y
%
Y      = ystar.*(ystar>0);
%
%      Construct OLS Estimates for the mis-specified model
%
thetahat = inv(X'*X)*(X'*y);           %OLS estimates
fiterr    = y - X*thetahat;           %OLS residuals
sighat    = sqrt((fiterr'*fiterr)/(nobs - size(X,2))); %sigma estimate
avarcov   = (sighat*inv(X'*X));       %variance-covariance matrix
stderrs   = sqrt(diag(sighat*inv(X'*X))); %standard errors
disp('Estimates of beta0, beta1 and sigma ');
[thetahat(1,1) thetahat(2,1) sighat]

disp('Estimate of asymptotic var-cov matrix');
avarcov

%
%      Simulate 90% Confidence interval
%
thetar   = repmat(thetahat,1,R) + chol(avarcov)'*randn(2,R);
gdraw    = sort((thetar(1,:)'+ 5*thetar(2,:))'.^2);
lbound   = gdraw(floor(R*0.05),1);
hbound   = gdraw(ceil(R*0.95),1);
disp('Estimates of lower and upper bounds of 90% confidence interval');
[lbound hbound]
figure(1);
hist(gdraw,hspace)
title('gdraws', 'FontSize', 12);

%
```

```

%      Part f: Repeat OLS estimation for R=1000 times and summarize
%      outcomes
%
thetahatr = zeros(2,R);
for      r      = 1:R;
    epsr      = sigma*randn(nobs,1);
    Xr        = [ones(nobs,1) rand(nobs,1)];
    ystarr    = Xr*theta + epsr;
    yr        = ystarr.*(ystarr>0);
    thetahatr(:,r) = inv(Xr'*Xr)*(Xr'*yr);           %OLS estimates
end;
%
%      Provide summary statistics for parameter estimates
%
disp('Mean, std, min and max for beta0');
[mean(thetahatr(1,:)) std(thetahatr(1,:)) min(thetahatr(1,:))...
 max(thetahatr(1,:))]

disp('Mean, std, min and max for beta1');
[mean(thetahatr(2,:)) std(thetahatr(2,:)) min(thetahatr(2,:))...
 max(thetahatr(2,:))]

%
%      Provide histograms for parameter estimates
%
figure(2);
subplot(2,1,1);
hist(thetahatr(1,:),hspace)
title('beta0', 'FontSize', 12);

subplot(2,1,2);
hist(thetahatr(2,:),hspace)
title('beta1', 'FontSize', 12);

%
%      Part g: Repeat part f with beta0=2;
%
thetag = [2 beta1]';
thetahatr = zeros(2,R);
for      r      = 1:R;
    epsr      = sigma*randn(nobs,1);
    Xr        = [ones(nobs,1) rand(nobs,1)];
    ystarr    = Xr*thetag + epsr;
    yr        = ystarr.*(ystarr>0);
    thetahatr(:,r) = inv(Xr'*Xr)*(Xr'*yr);           %OLS estimates
end;
%
%      Provide summary statistics for parameter estimates
%
disp('Mean, std, min and max for beta0');
[mean(thetahatr(1,:)) std(thetahatr(1,:)) min(thetahatr(1,:))...
 max(thetahatr(1,:))]

disp('Mean, std, min and max for beta1');
[mean(thetahatr(2,:)) std(thetahatr(2,:)) min(thetahatr(2,:))...
 max(thetahatr(2,:))]

%
%      Provide histograms for parameter estimates
%
figure(3);
subplot(2,1,1);
hist(thetahatr(1,:),hspace)

```

```

title('beta0', 'FontSize', 12);

subplot(2,1,2);
hist(thetahatr(2,:),hspace)
title('beta1', 'FontSize', 12);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%       Problem 2
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%       Draw from the truncated normal distribution TN[a,b](mu,sig^2)
%       where a   denotes the lower bound of truncation
%             b   denotes the upper bound of truncation
%             mu  denotes the mean of the untruncate distribution
%             sig denotes the standard deviation of the untruncated
%             distribution
%
a1   = 0;
b1   = 60;
mu   = 0;
sig  = 25;
ua1  = normcdf((a1 - mu)/sig);
ub1  = normcdf((b1 - mu)/sig);
u1   = ua1 + (ub1 - ua1)*rand(R2,1);
V1   = mu + sig*norminv(u1);
%
%       Draw from the discrete distribution (P1,P2,P3,P4,P5) where
%       Pi denotes the probability that V4=i
%
p     = cumsum([0.20 0.40 0.06 0.30 0.04]);
V2   = zeros(R2,1);
for i=1:R2;
    I=find(rand<p);
    V2(i,1)=min(I);
end;
%
%       Draw from the truncated extreme value distribution
%
a3   = 0.5;
b3   = 1.5;
ua3  = exp(-exp(-a3));
ub3  = exp(-exp(-b3));
u3   = ua3 + (ub3 - ua3)*rand(R2,1);
V3   = -log(-log(u3));
%
%       Use the acceptance/rejection method to draw from a triangular
%       distribution
%
a4   = 0.5;
b4   = 1.5;
a    = 2;
V4   = [];
while size(V4,1)<R2;
    z4   = (a4-b4) + 2*b4*rand(1,1);
    h4   = 0.5*b4;
    u4   = rand(1,1);
    if   z4<a4;
        f = (z4+b4-a4)/(b4^2);
    else;
        f = (a4+b4-z4)/(b4^2);
    end;
end;

```

```
        if      u4 < f/(a*h4);  
            V4      = [V4; z4];  
        end;  
end;  
%  
%      Create histograms for generated draws  
%  
figure(4);  
subplot(2,2,1);  
hist(V1,hspace)  
title('V1', 'FontSize', 12);  
  
subplot(2,2,2);  
hist(V2,hspace)  
title('V2', 'FontSize', 12);  
  
subplot(2,2,3);  
hist(V3,hspace)  
title('V3', 'FontSize', 12);  
  
subplot(2,2,4);  
hist(V4,hspace)  
title('V4', 'FontSize', 12);
```