November 26, 2018

Empirical Methods HA 5

Konstantin Guryev

Pennsylvania State University

2018

Problem 1 Log-likelihood is equal to -1257.1

Problem 2 Log-likelihood is equal to -1259.8

Problem 3 See the code.

Problem 4 See the code.

Problem 5 Starting values for estimation in Problem 3 : $(2; 1; 0)$;

For GQ-20 $(\hat{\beta}_0, \hat{\sigma}_\beta, \hat{\gamma}) = (2.4823; 1.4055; -0.5056)$;

GQ-20 maximal value of Log-likelihood is equal to 536.2378;

For MC-100 $(\hat{\beta}_0, \hat{\sigma}_\beta, \hat{\gamma}) = (2.5140; 1.2972; -0.5029)$;

MC-100 maximal value of Log-likelihood is equal to 538.3832;

Starting values for estimation in Problem 4 : $(3; 1, 5; 2; 1; 2; -0.5)$;

For MC-100 $(\hat{\beta}_0, \hat{u}_0, \hat{\sigma}_\beta, \hat{\sigma}_{\beta u}, \hat{\sigma}_u \hat{\gamma}) = (3.2223; 1.4286; 1.9100; 0.6473; 1.8390; -0.6929)$;

MC-100 maximal value of Log-likelihood is equal to 462.0722;

Matlab Code

```
clear;
load('hw5.mat')
rng(2);
beta0=0.1;
sig_beta=1;

F=@(eps) (1+exp(-eps)).^(-1);
% Likelihoods in vector form
LH_i=@(b,g,u) ...
    prod((F(data.X*diag(b)+data.Z*diag(g)+ones(20,100)*diag(u)).^
        data.Y).*...
    (1-F(data.X*diag(b)+data.Z*diag(g)+ones(20,100)*diag(u))).^(1-
        data.Y));

%% Problem 1
global x20 w20
[x20,w20]=GaussHermite_2(20);

int1=zeros(1,data.N);
for k=1:20
    int1=int1+w20(k)*LH_i(beta0+sqrt(2*sig_beta)*x20(k),0,0)/sqrt(pi
        );
end
% Log-likelihood
LLH1=sum(log(int1));

%% Problem 2

U=normrnd(beta0,sqrt(sig_beta),100,100);

int2=zeros(1,data.N);
for k=1:100
    int2=int2+(1/100)*LH_i(U(k,:),0,0);
end
```

```
% Log-likelihood
LLH2=sum(log(int2));

%% Problem 3

st_val1 =[2;1;0];
% GQ
fun1=@(x) -LLH_1(data.N,x(1),x(2),x(3),LH_i,1);
[estimate1,function1]=fmincon(fun1,st_val1,diag([0,-1,0]),zeros(3,1));

% MC
fun2=@(x) -LLH_1(data.N,x(1),x(2),x(3),LH_i,2);
[estimate2,function2]=fmincon(fun2,st_val1,diag([0,-1,0]),zeros(3,1));

%% Problem 4
fun3=@(x) -LLH_2(data.N,x(1),x(2),x(3),x(4),x(5),x(6),LH_i);


st_val2 =[3;1.5;2;1;2;-0.5];
A=diag([0,0,-1,0,-1,0]);
b=zeros(6,1);
[estimate3,function3]=fmincon(fun3,st_val2,A,b,[],[],[],[],@PSD);


%% Problem 5
display(LLH1,'log-likelihood GQ 20')
display(LLH2,'log-likelihood MC 100')
display(st_val1,'starting value for estimation in 3')
display(estimate1,'GQ 20 estimates of beta0,sigma_beta,gamma')
display(function1,'GQ 20 max val of LLH')
display(estimate2,'MC 100 estimates of beta0,sigma_beta,gamma')
display(function2,'MC 100 max val of LLH')
display(st_val2,'starting value for estimation in 4')
```

```
display(estimate3,'MC 100 estimates of beta0,u0,s_b,s_bu,su,gamma')
display(function3,'MC 100 max val of LLH')
```

Matlab Code

```matlab
function [x, w] = GaussHermite_2(n)

% This function determines the abscisas (x) and weights (w) for the
% Gauss-Hermite quadrature of order n>1, on the interval [-INF, +INF].
    % This function is valid for any degree n>=2, as the companion matrix
    % (of the n'th degree Hermite polynomial) is constructed as a
    % symmetrical matrix, guaranteeing that all the eigenvalues (roots)
    % will be real.


%  Geert Van Damme
% geert@vandamme-iliano.be
% February 21, 2010



% Building the companion matrix CM
    % CM is such that det(xI-CM)=L_n(x), with L_n the Hermite polynomial
    % under consideration. Moreover, CM will be constructed in such a way
    % that it is symmetrical.
i   = 1:n-1;
a   = sqrt(i/2);
CM  = diag(a,1) + diag(a,-1);

% Determining the abscissas (x) and weights (w)
    % - since det(xI-CM)=L_n(x), the abscissas are the roots of the
    %   characteristic polynomial, i.d. the eigenvalues of CM;
    % - the weights can be derived from the corresponding eigenvectors.
```

```
[V L]     = eig(CM);
[x ind] = sort(diag(L));
V         = V(:,ind)';
w         = sqrt(pi) * V(:,1).^2;
```

Matlab Code

```
function llh = LLH_1(N,b0,s_b,g,LH_i,method)

% Method corresponds to MC and GQ
rng(2);
global x20 w20
x=x20;
w=w20;

if method==2 % MC method
    U=normrnd(b0,sqrt(s_b),100,100);
    int=zeros(1,N);
    for k=1:100
        int=int+(1/100)*LH_i(U(k,:),g,0);
    end
    llh=sum(log(int));
else % GQ method
    int=zeros(1,N);
    for k=1:20
        int=int+w(k)*LH_i(b0+sqrt(2*s_b)*x(k),g,0)/sqrt(pi);
    end
    llh=sum(log(int));
end
```

Matlab Code

```
function llh = LLH_2(N,b0,u0,s_b,s_bu,s_u,g,LH_i)

% Here we use Monte Carlo method with 100 nodes

rng(2);
Mu=[b0;u0];

for t=1:100
    if s_b>=0 && s_u>=0 && s_b*s_u-s_bu^2>=0
        Sigma=[s_b,s_bu;s_bu,s_u];
        break
    else
        s_bu=sqrt(0.5*abs(s_b*s_u));
        s_b=s_b*(s_b>=0)+1*(s_b<0);
        s_u=s_u*(s_u>=0)+1*(s_u<0);
        Sigma=[1,s_bu;s_bu,1];
    end
end


X=(mvnrnd(Mu,Sigma,100*100))';
int3=zeros(1,N);
for k=1:100
    int3=int3+(1/100)*...
        LH_i(X(1,(k-1)*100+1:k*100),g,X(2,(k-1)*100+1:k*100));
end
llh=sum(log(int3));
```

Matlab Code

```
function [c, ceq] = PSD(x)

% Check on PSD of 2-by-2 x matrix

ceq=[];
V=[x(3),x(4);x(4),x(5)]
c=-det([x(3),x(4);x(4),x(5)]);
end
```