Homework 5

Kensuke Suzuki

November 23, 2018

Problem 1: Gaussian Quadrature

In this problem, we rule out u_i from the model. Define the function `llk_wou(Y,X,Z,par,node,method)` which returns (negative of) the log likelihood, given data (`X`, `Y`, and `Z`), parameter vector (`par`), number of node (`node`), and specified integration method (`method`).
In the first problem, I use the Gaussian quadrature method; `method=1` with 20 nodes. Using `qnwnorm( )` included in the CEtools, we draw 20 nodes for β_i from the normal distribution with mean β_0 and variance σ_β^2. This also generates the weighting vector **w** which I use later. I pick each draw of β_i, compute the the likelihood for each i, $L_i(\gamma|\beta_i, u_i)$, and stack it up for all draws. Numerical integration is completed by calculating the weighted average of the likelihood using the weights obtained above. Finally take log and sum over all i. Log likelihood is $-1.2571e+03$.

Problem 2: Monte Carlo

In the second problem, I use the Monte Carlo method; `method=2` with 100 nodes. I draw 100 nodes using `haltonNormshuddle( )` provided in the lecture. Analogous to the first problem, for each draw, we compute the likelihood $L_i(\gamma|\beta_i, u_i)$, stack it up for all draws, and compute the simple average. Finally take log and sum over all i. Log likelihood is $-1.2571e+03$.

Problem 3: MLE without u_i using `fminsearch`

We use `fminsearch` to estimate the parameters. Use the same function explained above. Results are presented below:

Gaussian Quadrature

$$\text{Initial guess: } \begin{bmatrix} \gamma \\ \beta_0 \\ \sigma_\beta \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \quad \text{estimates: } \begin{bmatrix} \gamma \\ \beta_0 \\ \sigma_\beta \end{bmatrix} = \begin{bmatrix} -0.5056 \\ 2.4832 \\ 1.4054 \end{bmatrix}, \quad \text{loglikelihood: } -536.2378$$

Monte Carlo

$$\text{Initial guess: } \begin{bmatrix}\gamma\\ \beta_0\\ \sigma_\beta\end{bmatrix} = \begin{bmatrix}1\\1\\1\end{bmatrix}, \quad \text{estimates: } \begin{bmatrix}\gamma\\ \beta_0\\ \sigma_\beta\end{bmatrix} = \begin{bmatrix}-0.5056\\ 2.5578\\ 1.1816\end{bmatrix}, \quad \text{loglikelihood: } -536.5876$$

Matlab function llk_wou( )

```
function [llf,methodname] = llk_wou(Y,X,Z,par,node,method)
    % compute negative of llf

gamma = par(1);
betanot = par(2);
sigmab = par(3);
%unot = par(4);
unot = 0;
%sigmaub = par(5);
sigmaub = 0;
%sigmau = par(6);
sigmau = 0;

mu = [betanot unot];
Sigma = [sigmab sigmaub; sigmaub sigmau];

ui = 0;


if method == 1
    methodname = 'Gaussian Quadrature';
        % if method is Gaussian Quadrature
    [rcoef,w] = qnwnorm(node, betanot, sigmab);


for i = 1:length(rcoef)
    betai = rcoef(i,1);
        % pick ith draw of beta
    epsi = (betai * X + gamma * Z + ui  ) ;
    logitval = ( 1 + exp(-1 * epsi) ).^(-1);
        % compute the logistic CDF
    lkt = logitval.^Y .* (ones(20,100)-logitval).^(ones(20,100)-Y);
        % compute the contribution of each year
    lkii(i,:) = prod(lkt);
        % product over years
end
lki = w' * lkii;
    % numerical integration

```

```
llki = log(lki);

llf = -1 * sum(llki,2);

elseif method == 2
    methodname = 'Monte Carlo';
        % if method is MC

    norm = haltonNormShuffle(node, 1, 3);
    rcoef = repmat(betanot,node,1) + sigmab * norm';

for i = 1:length(rcoef)
    betai = rcoef(i,1);
        % pick ith draw of beta
    epsi = (betai * X + gamma * Z + ui  ) ;
    logitval = ( 1 + exp(-1 * epsi) ).^(-1);
        % compute the logistic CDF
    lkt = logitval.^Y .* (ones(20,100)-logitval).^(ones(20,100)-Y);
        % compute the contribution of each year
    lkii(i,:) = prod(lkt);
        % product over years
end
lki = sum(1/node * lkii);
    % numerical integration

llki = log(lki);

llf = -1 * sum(llki,2);

end


end
```

Problem 4: MLE of full model using `fminsearch`

In this problem, we estimate the full model. Define the function `llk_wu(Y,X,Z,par,node,method)` which returns (negative of) the log likelihood of the full model, given data (`X`, `Y`, and `Z`), parameter vector (`par`), number of node (`node`), and specified integration method (`method`). Since we only invoke Monte Carlo method, `method=2`.
In this function, I draw 100 nodes using `haltonNormshuddle( )`. For this time, I draw β_i and u_i from the bivariate normal distribution with mean $\mu = [\beta_0, u_0]'$ and variance-covariance matrix $\Sigma = \begin{bmatrix} \sigma_\beta & \sigma_{u\beta} \\ \sigma_{u\beta} & \sigma_u \end{bmatrix}$. I use `chol( )` to make Cholesky decomposition of Σ to simulate from the joint distribution—bivariate normal. Implementation of numerical integration is same as in Problem 2. For optimization, we use `fminsearch` same as before. Results are presented below:

$$\text{Initial guess: } \begin{bmatrix} \gamma \\ \beta_0 \\ u_0 \\ \sigma_\beta^2 \\ \sigma_{ub} \\ \sigma_u^2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0.5 \\ 1 \end{bmatrix}, \quad \text{estimates: } \begin{bmatrix} \gamma \\ \beta_0 \\ u_0 \\ \sigma_\beta^2 \\ \sigma_{ub} \\ \sigma_u^2 \end{bmatrix} = \begin{bmatrix} -0.6815 \\ 3.1877 \\ 1.4710 \\ 1.9226 \\ 0.8068 \\ 1.6458 \end{bmatrix}, \quad \text{loglikelihood: } -464.0001$$

Matlab function `llk_wu( )`

```
function [llf ,methodname] = llk_wu(Y,X,Z,par ,node ,method)
    % compute negative of llf

gamma = par(1);
betanot = par(2);
sigmab = par(3);
unot = par(4);
sigmaub = par(5);
sigmau = par(6);


mu = [betanot unot];
Sigma = [sigmab sigmaub; sigmaub sigmau];
U = chol(Sigma);

if method == 2
    methodname = 'Monte Carlo';
        % if method is MC

    norm = haltonNormShuffle(node, 2, 2);
    rcoef = repmat(mu,node, 1) + (U' * norm)';

for i = 1:length(rcoef)
    betai = rcoef(i,1);
    ui = rcoef(i,2);
        % pick ith draw of beta
    epsi = (betai * X + gamma * Z + ui   ) ;
    logitval = ( 1 + exp(-1 * epsi) ).^(-1);
        % compute the logistic CDF
    lkt = logitval.^Y .* (ones(20,100)-logitval).^(ones(20,100)-Y);
        % compute the contribution of each year
    lkii(i,:) = prod(lkt);
        % product over years
end
lki = sum(1/node * lkii);
    % numerical integration

```

```
llki = log(lki);

llf = -1 * sum(llki,2);

end


end
```

Matlab Main Code

```
% Empirical Method HW5 %
% Ken Suzuki (Penn State)
% kxs974@psu.edu

clear all
delete HW5log.txt
diary('HW5log.txt')
diary on

% Load Data
load('hw5.mat')

X = data.X;
Y = data.Y;
Z = data.Z;

addpath('../CEtools/');

%% Problem 1
% parameter value
betanot = 0.1;
sigmab = 1;
gamma = 0;

% method
method = 1;

% number of nodes
node = 20;

% set parameter vector
par = [gamma betanot sigmab];

[llf,methodname] = llk_wou(Y,X,Z,par,node,method);

llf = -1 * llf;
% display result
```

```
disp('Problem 1')
disp(methodname)
disp('Loglikelihood is')
disp(llf)

%% Problem 2
% parameter value
betanot = 0.1;
sigmab = 1;
gamma = 0;

% method
method = 2;

% number of nodes
node = 100;

% set parameter vector
par = [gamma betanot sigmab];

[llf,methodname] = llk_wou(Y,X,Z,par,node,method);
llf = -1 * llf; % take negative

% display result
disp('Problem 2')
disp(methodname)
disp('Loglikelihood is')
disp(llf)

%% Problem 3

clear par

% number of node
node = 20;

% method: GC
method = 1;

% define function to be minimzied (function of par)
llkwou_min = @(par) llk_wou(Y,X,Z,par,node,method);

% fminsearch
[paraGQ, lfGQ] = fminsearch(llkwou_min, [1 1 1] );
lfGQ = -1 * lfGQ;

% number of node
node = 100;
```

```
% method: GC
method = 2;

% define function to be minimzied (function of par)
llkwou_min = @(par) llk_wou(Y,X,Z,par,node,method);

% fminsearch
[paraMC, lfMC] = fminsearch(llkwou_min, [1 1 1] );
lfMC = -1 * lfMC;


% display result
disp('Problem 3-1 (Gaussian Quadrature)')
disp('Initial guesses are')
disp('    gamma       beta      sigmab')
disp([1 1 1])
disp('    gamma       beta      sigmab')
disp(paraGQ)
disp('Maximized log-likelihood is:')
disp(lfGQ)


% display result
disp('Problem 3-2 (Monte Carlo)')
disp('Initial guesses are')
disp('    gamma       beta      sigmab')
disp([1 1 1])
disp('    gamma       beta      sigmab')
disp(paraMC)
disp('Maximized log-likelihood is:')
disp(lfMC)


%% Problem 4

% method: MC
method = 2;

% number of node
node = 100;

% initial values for parameter vector
gamma = 1;
betanot = 1;
sigmab = 1;
unot = 1;
sigmaub = 0.5;
```

```
sigmau =1;
intpar = [gamma betanot sigmab unot sigmaub sigmau];

%define function to be minimized
llkwu_min = @(par) llk_wu(Y,X,Z,par,node,method);

% fminsearch
[paraMC, lfMC] = fminsearch(llkwu_min, intpar );
lfMC = -1 * lfMC;

% display result
disp('Problem 4 (Monte Carlo)')
disp('Initial guesses are')
disp('    gamma       betanot    sigmab     unot        sigmaub    sigmau')
disp(intpar)
disp('Estimated parameters')
disp('    gamma       betanot    sigmab     unot        sigmaub    sigmau')
disp(paraMC)
disp('Maximized log-likelihood is:')
disp(lfMC)

diary off
```