

# 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  $\beta_i$  from the normal distribution with mean  $\beta_0$  and variance  $\sigma_\beta^2$ . This also generates the weighting vector `w` which I use later. I pick each draw of  $\beta_i$ , 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

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

**Monte Carlo**

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

**Matlab function** `llk_wou( )`

```

1 function [llf,methodname] = llk_wou(Y,X,Z,par,node,method)
2     % compute negative of llf
3
4     gamma = par(1);
5     betanot = par(2);
6     sigmab = par(3);
7     %unot = par(4);
8     unot = 0;
9     %sigmaub = par(5);
10    sigmaub = 0;
11    %sigmau = par(6);
12    sigmau = 0;
13
14    mu = [betanot unot];
15    Sigma = [sigmab sigmaub; sigmaub sigmau];
16
17    ui = 0;
18
19
20    if method == 1
21        methodname = 'Gaussian Quadrature';
22        % if method is Gaussian Quadrature
23        [rcoef,w] = qnwnorm(node, betanot, sigmab);
24
25
26    for i = 1:length(rcoef)
27        betai = rcoef(i,1);
28        % pick ith draw of beta
29        epsi = (betai * X + gamma * Z + ui) ;
30        logitval = ( 1 + exp(-1 * epsi) ).^(-1);
31        % compute the logistic CDF
32        lkt = logitval.^Y .* (ones(20,100)-logitval).^(ones(20,100)-Y);
33        % compute the contribution of each year
34        lkii(i,:) = prod(lkt);
35        % product over years
36    end
37    lki = w' * lkii;
38    % numerical integration
39

```

```

40 llki = log(lki);
41
42 llf = -1 * sum(llki,2);
43
44 elseif method == 2
45     methodname = 'Monte Carlo';
46     % if method is MC
47
48     norm = haltonNormShuffle(node, 1, 3);
49     rcoef = repmat(betanot,node,1) + sigmab * norm';
50
51 for i = 1:length(rcoef)
52     betai = rcoef(i,1);
53     % pick ith draw of beta
54     epsi = (betai * X + gamma * Z + ui ) ;
55     logitval = ( 1 + exp(-1 * epsi) ).^(-1);
56     % compute the logistic CDF
57     lkt = logitval.^Y .* (ones(20,100)-logitval).^(ones(20,100)-Y);
58     % compute the contribution of each year
59     lkii(i,:) = prod(lkt);
60     % product over years
61 end
62 lki = sum(1/node * lkii);
63 % numerical integration
64
65 llki = log(lki);
66
67 llf = -1 * sum(llki,2);
68
69 end
70
71
72 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  $\beta_i$  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  $\Sigma$  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( )`

```

1 function [llf,methodname] = llk_wu(Y,X,Z,par,node,method)
2     % compute negative of llf
3
4     gamma = par(1);
5     betanot = par(2);
6     sigmab = par(3);
7     unot = par(4);
8     sigmaub = par(5);
9     sigmau = par(6);
10
11
12     mu = [betanot unot];
13     Sigma = [sigmab sigmaub; sigmaub sigmau];
14     U = chol(Sigma);
15
16     if method == 2
17         methodname = 'Monte Carlo';
18         % if method is MC
19
20         norm = haltonNormShuffle(node, 2, 2);
21         rcoef = repmat(mu,node, 1) + (U' * norm)';
22
23     for i = 1:length(rcoef)
24         betai = rcoef(i,1);
25         ui = rcoef(i,2);
26         % pick ith draw of beta
27         epsi = (betai * X + gamma * Z + ui) ;
28         logitval = ( 1 + exp(-1 * epsi) ).^(-1);
29         % compute the logistic CDF
30         lkt = logitval.^Y .* (ones(20,100)-logitval).^(ones(20,100)-Y);
31         % compute the contribution of each year
32         lkii(i,:) = prod(lkt);
33         % product over years
34     end
35     lki = sum(1/node * lkii);
36     % numerical integration
37

```

```
38 llki = log(lki);
39
40 llf = -1 * sum(llki,2);
41
42 end
43
44
45 end
```

## Matlab Main Code

```
1 % Empirical Method HW5 %
2 % Ken Suzuki (Penn State)
3 % kxs974@psu.edu
4
5 clear all
6 delete HW5log.txt
7 diary('HW5log.txt')
8 diary on
9
10 % Load Data
11 load('hw5.mat')
12
13 X = data.X;
14 Y = data.Y;
15 Z = data.Z;
16
17 addpath(' ../ CTools/ ');
18
19 %% Problem 1
20 % parameter value
21 betanot = 0.1;
22 sigmab = 1;
23 gamma = 0;
24
25 % method
26 method = 1;
27
28 % number of nodes
29 node = 20;
30
31 % set parameter vector
32 par = [gamma betanot sigmab];
33
34 [llf,methodname] = llk_wou(Y,X,Z,par,node,method);
35
36 llf = -1 * llf;
37 % display result
```

```
38 disp('Problem 1')
39 disp(methodname)
40 disp('Loglikelihood is')
41 disp(llf)
42
43 %% Problem 2
44 % parameter value
45 betanot = 0.1;
46 sigmab = 1;
47 gamma = 0;
48
49 % method
50 method = 2;
51
52 % number of nodes
53 node = 100;
54
55 % set parameter vector
56 par = [gamma betanot sigmab];
57
58 [llf,methodname] = llk_wou(Y,X,Z,par,node,method);
59 llf = -1 * llf; % take negative
60
61 % display result
62 disp('Problem 2')
63 disp(methodname)
64 disp('Loglikelihood is')
65 disp(llf)
66
67 %% Problem 3
68
69 clear par
70
71 % number of node
72 node = 20;
73
74 % method: GC
75 method = 1;
76
77 % define function to be minimized (function of par)
78 llkwou_min = @(par) llk_wou(Y,X,Z,par,node,method);
79
80 % fminsearch
81 [paraGQ, lfGQ] = fminsearch(llkwou_min, [1 1 1]);
82 lfGQ = -1 * lfGQ;
83
84 % number of node
85 node = 100;
```

```
86
87 % method: GC
88 method = 2;
89
90 % define function to be minimized (function of par)
91 llkwou_min = @(par) llkwou(Y,X,Z,par,node,method);
92
93 % fminsearch
94 [paraMC, lfMC] = fminsearch(llkwou_min, [1 1 1] );
95 lfMC = -1 * lfMC;
96
97
98 % display result
99 disp('Problem 3-1 (Gaussian Quadrature)')
100 disp('Initial guesses are')
101 disp('    gamma    beta    sigmab')
102 disp([1 1 1])
103 disp('    gamma    beta    sigmab')
104 disp(paraGQ)
105 disp('Maximized log-likelihood is:')
106 disp(lfGQ)
107
108
109 % display result
110 disp('Problem 3-2 (Monte Carlo)')
111 disp('Initial guesses are')
112 disp('    gamma    beta    sigmab')
113 disp([1 1 1])
114 disp('    gamma    beta    sigmab')
115 disp(paraMC)
116 disp('Maximized log-likelihood is:')
117 disp(lfMC)
118
119
120 %% Problem 4
121
122 % method: MC
123 method = 2;
124
125 % number of node
126 node = 100;
127
128 % initial values for parameter vector
129 gamma = 1;
130 betanot = 1;
131 sigmab = 1;
132 unot = 1;
133 sigmaub = 0.5;
```

```
134 sigmau =1;
135 intpar = [gamma betanot sigmab unot sigmaub sigmau];
136
137 %define function to be minimized
138 llkwu_min = @(par) llkwu(Y,X,Z,par,node,method);
139
140 % fminsearch
141 [paraMC, lfMC] = fminsearch(llkwu_min, intpar );
142 lfMC = -1 * lfMC;
143
144 % display result
145 disp('Problem 4 (Monte Carlo)')
146 disp('Initial guesses are')
147 disp('    gamma    betanot    sigmab    unot    sigmaub    sigmau')
148 disp(intpar)
149 disp('Estimated parameters')
150 disp('    gamma    betanot    sigmab    unot    sigmaub    sigmau')
151 disp(paraMC)
152 disp('Maximized log-likelihood is:')
153 disp(lfMC)
154
155 diary off
```