

# Homework 2

Kensuke Suzuki

September 25, 2018

The elapsed times described below may not coincide with the ones in the log file—They may change every time I run the command, but the relative size between algorithms does not change.

## Problem 1

I define a function `bertrand.m` which returns vector of demand for each good, for given vector of price  $\mathbf{p}$  and  $\mathbf{v}$  (which are potentially  $n \times 1$  vectors).

```
1 function fval = bertrand(p,v)
2
3 % for given vector of p and v, solve demand
4 fval = exp(v - p) ./ ( 1 + sum( exp(v - p) ) );
5
6 end
```

For  $\mathbf{p} = [1, 1]^\top$  and  $\mathbf{v} = [2, 2]^\top$ , we obtain  $[D_A, D_B]^\top = [0.422319, 0.422319]^\top$  and  $D_0 = 1.55362$ .

## Problem 2

Each firm solves profit maximization problem:

$$\max_{p_i} p_i D_i$$

for  $i = A, B$ . The first order conditions yields:

$$D_i + \left[ \frac{\partial D_i}{\partial p_i} p_i \right] = D_i - p_i D_i (1 - D_i) = D_i [1 - p_i (1 - D_i)] = 0$$

Provided  $D_i \neq 0$ , the FOC boils down to  $[1 - p_i(1 - D_i)] = 0$ . The Bertrand-Nash equilibrium is the set of prices which satisfy the system of equation:

$$\begin{aligned} 1 - p_A(1 - D_A) &= 0 \\ 1 - p_B(1 - D_B) &= 0 \end{aligned}$$

In matrix notation, we have

$$\begin{bmatrix} 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 1 - D_A & 0 \\ 0 & 1 - D_B \end{bmatrix} \begin{bmatrix} p_A \\ p_B \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad (1)$$

We define the LHS of (1) as a function `bertrandfoc.m`

```
1 function fval = bertrandfoc(p,v)
2
3 % for given vector of p and v, solve demand
4 D = exp(v - p) ./ ( 1 + sum( exp(v - p) ) );
5
6 % First order condition boils down to
7 fval = ones(size(p,1),1) - diag(ones(size(p,1),1)-D)*p;
8
9 end
```

We then solve the system of nonlinear equation using Broyden's Method. The algorithm is completely analogue to what we did in the class. For the starting value of  $\mathbf{p}$ , we use  $[1, 1]^\top$  and we use the identity matrix as an initial inverse of Jacobian. For convergence criterion, we use  $1e-6$ . The iteration converges and we get the set of equilibrium prices  $\mathbf{p} = [1.598942, 1.598942]^\top$

### Problem 3

First we define the function `bertrandfocg.m` which return the FOC for  $g$ th good price.

```
1 function fval = bertrandfocg(p,v,g)
2
3 % for given vector of p and v, solve demand
4 D = exp(v - p) ./ ( 1 + sum( exp(v - p) ) );
5
6 % First order condition boils down to
7 foc = ones(size(p,1),1) - diag(ones(size(p,1),1)-D)*p;
8
9 fval = foc(g,1);
10
11 end
```

We then solve the system by using a Gauss-Seidel method. First we set the initial value for  $\mathbf{p} = [1, 1]^\top$  and set  $\mathbf{p}_{\text{old}} = [2, 2]$ . Then, for given  $p_B$  we solve the FOC for good  $A$  price using the secant method. This sub-iteration solves  $p_A$  for given initial guess on  $p_B$ . Then next sub-iteration solves  $p_B$  using the FOC for good  $B$  price using the  $p_A$  obtained in the previous sub-iteration.

$$\begin{aligned}(1) \quad p_A^{k+1} &\Leftarrow 1 - p_A^k(1 - D_A(p_A^k, p_B^k)) = 0 \\(2) \quad p_B^{k+1} &\Leftarrow 1 - p_B^k(1 - D_B(p_A^{k+1}, p_B^k)) = 0\end{aligned}$$

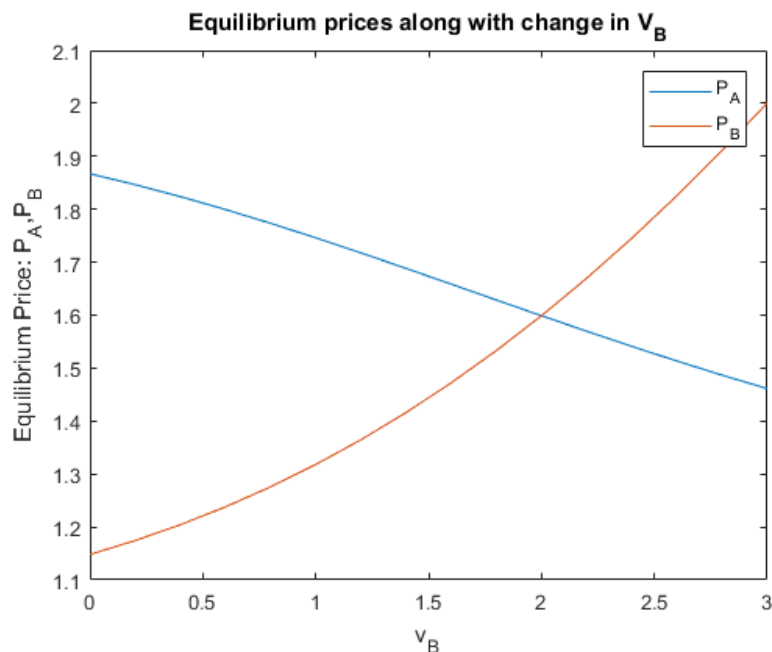
We iterate the set of two subiteration (indexed by  $k$  above) until the norm of the vector obtained using the function `bertrandfoc.m` (which returns the vector of the LHS of FOCs) is below the tolerance level. This algorithm also yields the same equilibrium price vector  $\mathbf{p} = [1.598942, 1.598942]^\top$ . The elapsed time for problem 2 (Broyden Method) is 0.005407 and the one for problem 3 (Gauss-Seidel Method) is 0.014861. Therefore, Gauss-Seidel Method is slower. **WHY:** intuitively it is because the Gauss-Seidel method does not update the  $p_A$  and  $p_B$  at once, rather, it solves two equations separately. More precisely, the Gauss-Seidel method solves for the first equation (FOC for firm A) for given price of  $p_B$  (step 1) and then solves for the second equation (FOC for firm B)  $p_B$  for given price of  $p_A$  (step 2). Each step entails iteration. Of course,  $p_A$  obtained in the first step is not necessarily the best response for given  $p_B$  obtained in the subsequent step. Therefore, we need to iterate this set of two steps until it converges. So there exists “double loop,” which make the algorithm slower than Broyden.

## Problem 4

In this problem, we use the update rule specified in the problem set and solve the system. It again yields the same result:  $\mathbf{p} = [1.598942, 1.598942]^\top$ . Time elapsed was 0.007433.

## Problem 5

We define the vector of  $v_B$  values (0:2:2) and for each  $v_B$  and  $v_A = 2$ , we solve the system of equation for  $\mathbf{p}$ . We store the equilibrium vector of prices in `result` matrix (where the first row is  $v_B$  and the second and third rows contain the vector of equilibrium prices corresponding to each  $v_B$  value). 2-way plotted graph is demonstrated below.



## Matlab Code

```

1 % ECON512 Homework 2
2 % Kensuke Suzuki
3 clear all
4 delete HW2log.txt
5 diary('HW2log.txt')
6 diary on
7
8 disp('ECON512 HOMEWORK2: Ken Suzuki')
9
10 disp(' ')
11
12 %% Define bertrand and bertrandfoc function
13 % bertrand: return demand for each good
14 % bertrandfoc: return system of FOC (LHS)
15
16 %% Problem 1
17
18 p = [1;1];
19 v = [2;2];
20
21 Ans1 = bertrand(p,v)
22
23 D0 = 1 / (1+ sum(exp(v-p)) );
24
25 P1 = sprintf('Problem1: for vA=vB=2 and pA=pB=1, DA= %f , DA= %f , and D0
    = %f. ', Ans1(1,1), Ans1(2,1), D0);

```

```
26 disp(P1);
27 disp(' ')
28
29
30 %% Problem 2
31
32 clear all
33
34 v = [2;2];
35
36 p = [1;1];
37
38 fVal_foc = @(p) bertrandfoc(p,v);
39 i_fVal_foc = fVal_foc(p);
40
41 iJac = eye(size(p,1));
42
43 maxit = 100;
44 tol = 1e-6;
45
46 tic
47 for iter = 1:maxit
48     fnorm = norm(i_fVal_foc);
49     fprintf('iter %d: p(1)=%f, p(2)=%f, norm(f(x))=%.8f \n', iter, p(1),
50           p(2),norm(i_fVal_foc));
51     if norm(i_fVal_foc)<tol
52         break
53     end
54     d = -(iJac*i_fVal_foc);
55     p = p + d;
56     fOld_foc = i_fVal_foc;
57     i_fVal_foc = fVal_foc(p);
58     u = iJac*(i_fVal_foc - fOld_foc);
59     iJac = iJac + ( (d-u)*(d'*iJac) )/(d'*u);
60 end
61 elapsedTime_p2 = toc;
62 P2 = sprintf('Problem2: for vA=vB=2, equilibrium prices are: PA= %f, PB
63           = %f; time elapsed is %f.', p(1,1),p(2,1), elapsedTime_p2);
64 disp(P2);
65 disp(' ')
66
67
68 %% Problem 3
69
70 clear all
71
```

```

72 v = [2;2];
73 p = [1;1];
74
75 fVal_foc = @(p) bertrandfoc(p,v);
76 fVal_focg = @(p,g) bertrandfocg(p,v,g);
77
78 maxit = 100;
79 tol = 1e-6;
80
81 tic
82 for iter = 1:maxit
83
84     fval = fVal_foc(p);
85     if norm(fval) < tol
86         break
87     end
88     fprintf('iter %d: p(1)=%f, p(2)=%f, norm(f(x))=%.8f \n', iter, p(1)
89           ,p(2),norm(fval));
90
91     % set pOld
92     pOld = [2;2];
93     pA_Old = pOld(1,1);
94
95     % compute the LHS of FOC for good A price
96     fOld_1 = fVal_focg(pOld,1);
97
98     % for given pB, solve the first equation for pA
99     % We use Secant Method
100     for iter_1 = 1:maxit
101         fval_1 = fVal_focg(p,1);
102         if abs(fval_1) < tol
103             break
104         else
105             pA_New = p(1,1) - ( (p(1,1) - pA_Old) / (fval_1 - fOld_1) )
106                 * fval_1;
107             pA_Old = p(1,1);
108             p(1,1) = pA_New;
109             fOld_1 = fval_1;
110         end
111     end
112
113     % Use the solution for pA obtained above, solve for pB
114     pB_Old = pOld(2,1);
115     fOld_2 = fVal_focg(pOld,2);
116     for iter_2 = 1:maxit
117         fval_2 = fVal_focg(p,2);
118         if abs(fval_2) < tol
119             break

```

```

118         else
119             pB_New = p(2,1) - ( (p(2,1) - pB_Old) / (fval_2 - fOld_2) )
120                 * fval_2;
121             pB_Old = p(2,1);
122             p(2,1) = pB_New;
123             fOld_2 = fval_2;
124         end
125     end
126 end
127 elapsedTime_p3 = toc;
128
129 P3 = sprintf('Problem3: for vA=vB=2, equilibrium prices are: PA= %f, PB
130             = %f; time elapsed is %f.', p(1,1), p(2,1), elapsedTime_p3);
131 disp(P3);
132 disp(' ')
133
134 %% Problem 4
135
136 clear all
137
138 v = [2;2];
139
140 p = [1;1];
141
142 fVal_bertrand = @(p) bertrand(p,v);
143 fVal_foc = @(p) bertrandfoc(p,v);
144 i_fVal_foc = fVal_foc(p);
145
146 maxit = 100;
147 tol = 1e-6;
148
149 tic
150 for iter = 1:maxit
151     fnorm = norm(i_fVal_foc);
152     fprintf('iter %d: p(1)=%f, p(2)=%f, norm(f(x))=%.8f \n', iter, p(1),
153           p(2), norm(i_fVal_foc));
154     if norm(i_fVal_foc) < tol
155         break
156     end
157     p_next = 1./ ([1;1] - fVal_bertrand(p) );
158     p = p_next;
159     i_fVal_foc = fVal_foc(p);
160 end
161 elapsedTime_p4 = toc;
162
163 P4 = sprintf('Problem4: for vA=vB=2, equilibrium prices are: PA= %f, PB

```

```

    = %f; time elapsed is %f.', p(1,1),p(2,1), elapsedTime_p4);
163 disp(P4);
164 disp(' ')
165
166
167 %% Problem 5
168
169 clear all
170
171 vB_5 = [0:.2:3];
172 v_5 = [2*ones(1, size(vB_5,2)); vB_5 ];
173 result = [vB_5; ones(1, size(vB_5,2)); ones(1, size(vB_5,2)) ];
174
175 for vindex = 1:size(vB_5,2)
176
177     p = [1;1];
178     v = v_5(:,vindex);
179
180     fVal_foc = @(p) bertrandfoc(p,v);
181     i_fVal_foc = fVal_foc(p);
182     iJac = eye(size(p,1));
183
184     maxit = 100;
185     tol = 1e-6;
186
187     for iter = 1:maxit
188         fnorm = norm(i_fVal_foc);
189         %fprintf('iter %d: p(1)=%f, p(2)=%f, norm(f(x))=%.8f \n', iter ,
190             p(1),p(2),norm(i_fVal_foc));
191         if norm(i_fVal_foc)<tol
192             break
193         end
194         d = -(iJac*i_fVal_foc);
195         p = p + d;
196         fOld_foc = i_fVal_foc;
197         i_fVal_foc = fVal_foc(p);
198         u = iJac*(i_fVal_foc - fOld_foc);
199         iJac = iJac + ( (d-u)* (d'*iJac) )/(d'*u);
200     end
201     result(2,vindex) = p(1);
202     result(3,vindex) = p(2);
203 end
204
205 plot(vB_5, result(2,:), vB_5, result(3,:))
206 title('Equilibrium prices along with change in V_B')
207 xlabel('v_B')
208 ylabel('Equilibrium Price: P_A,P_B')
209 legend('P_A', 'P_B')

```



209

210 `diary` off