1: The First Problem

I omit the copy of the diary file (HW3.out) in this file because it's too long and latex gives an error when include it in this file. HW3.out is in the same directory as this file.

From Question 1 to 4, I use $(\log(\bar{y}), 0, 0, 0, 0, 0)$ as the initial guess, where $\bar{y}$ is the mean of the vector y.

The estimate from the Nelder-Mead method for the maximum likelihood is
$\hat{\beta}_{NM.ML} = (2.5339, -0.0323, 0.1157, -0.3540, 0.0798, -0.4094)$.

2: The Second Problem

(a) When I write up the gradient of the objective function, the estimate from the BFGS method for the maximum likelihood (ML) is
$\hat{\beta}_{BFGS.ML.1} = (2.5339, -0.0323, 0.1157, -0.3540, 0.0798, -0.4094)$.

(b) When I dont' write up the gradient of the likelihood, the estimate is the same as the one from the BFGS method with the gradient for the ML, that is,
$\hat{\beta}_{BFGS.ML.2} = (2.5339 - 0.0323, 0.1157, -0.3540, 0.0798, -0.4094)$.

3: The Third Problem

The estimate from command lsqnonlin for the nonlinear least squares (NLS) is
$\hat{\beta}_{NLS1} = (0.3895, -0.0146, 0.1193, -0.1242, 0.0770, -0.1732)$.

4: The Fourth Problem

The estimate from the Nelder-Mead method for the NLS
is$\hat{\beta}_{NLS2} = (2.5126, -0.0384, 0.1141, -0.2796, 0.0676, -0.3698)$

5: The Fifth Question

I examine how the estimates react if I change the initial guess. The initial guess is
$(\log(\bar{y}), 0, a, 0, 0, 0)$, where a moves from the 1st element to the last element in [-10:0.5:10].

Figure 1 shows the estimates for the coefficient for the number of years married β_2. The horizontal axis shows a, that is, the values of the initial guess for β_2. We observe the sharp decline around -8 for the Nelder-Mead NLS.

Figure 2 shows the figure that removes the line for the Nelder-Mead NLS from Figure 1. Function lsqnonlin gives the plausible estimate for only a limited range of the initial guesses around 0 (the purple line).

Figure 3 shows the figure that removes the line for the lsqnonlin NLS. Up to -0.5, the estimate from the BFGS ML is more stable than that from the Nelder-Mead ML. But above 0.5, the estimate from the BFGS goes away from the seemingly decent estimate 0.12.

Figure 4 compares the estimate from the Nelder-Mead ML and those from the BFGS ML (with

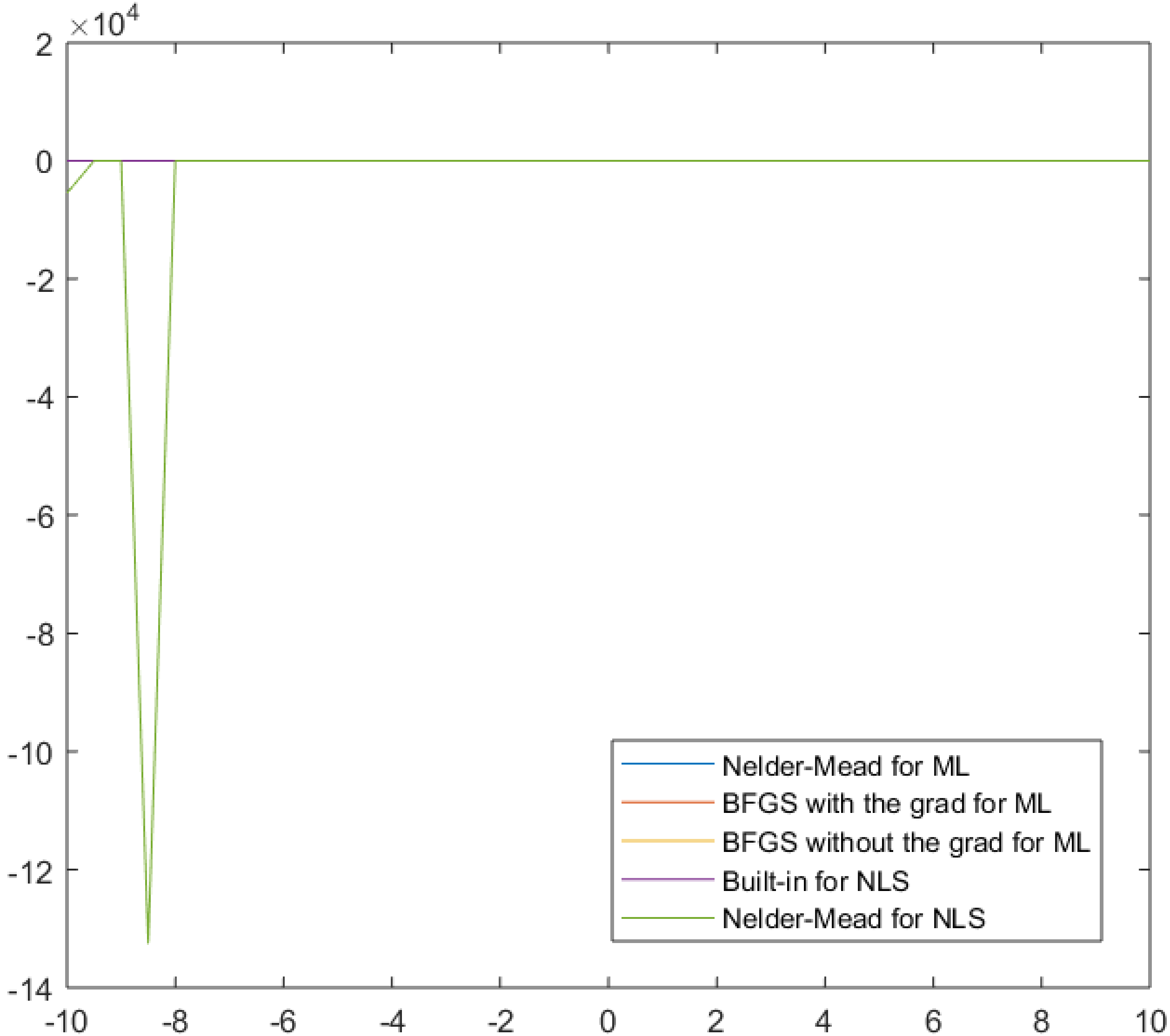

Figure 1: Comparison of All the 5

Table 1: Computation Time

Method	Time (Seconds)
Nelder-Mead, ML	1.756
BFGS with grad, ML	3.096
BFGS without grad, ML	0.969
lsqnonlin	2.179
Nelder-Mead for NLS	0.938

the analytical gradient). Now I see that the estimates from the BFGS with the analytical gradient for the ML and those from the BFGS without the analytical gradient are the same for the coefficient for the years married β_2.

Table 1 shows the computation time for each method. The 2nd column shows the time to compute all the estimates for the initial guesses $(\log(\bar{y}), 0, a, 0, 0, 0)$, where a takes values in [-10:0.5:10]. The BFGS without the analytical gradient was the fastest, and that with the analytical gradient was the slowest. The reason is that in my code in each step of the for loop, the function defining the likelihood and the gradient reads the data X, y. This is purely to avoid errors, for otherwise I got an error (when I defined a function for the data and beta and redefined the function for beta in the main file HW3.m).

Figure 5 compares the estimate for the coefficient for the constant β_0 from the Nelder-Mead ML and the BFGS ML. The BFGS is more stable locally.

Overall, the BFGS is better preferred to the ML by the Nelder-Mead for the local stability and the computation time. But as shown in Figure 4 shows, it does not work very well in some region of initial guesses. My ranking is: ML by BFGS>ML by Nelder Mead>NLS by lsqnonlin>NLS by Nelder Mead.

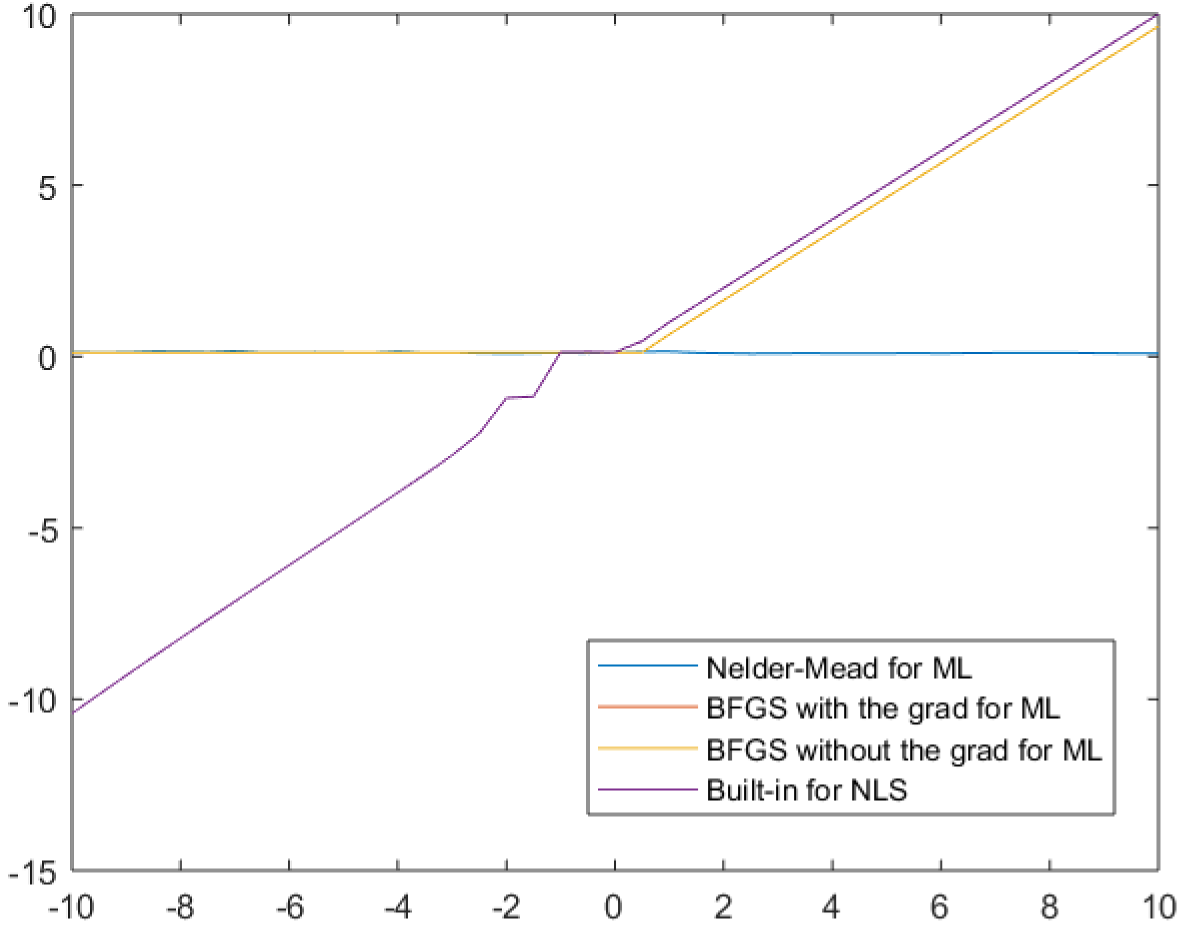

Figure 2: Comparison of 4

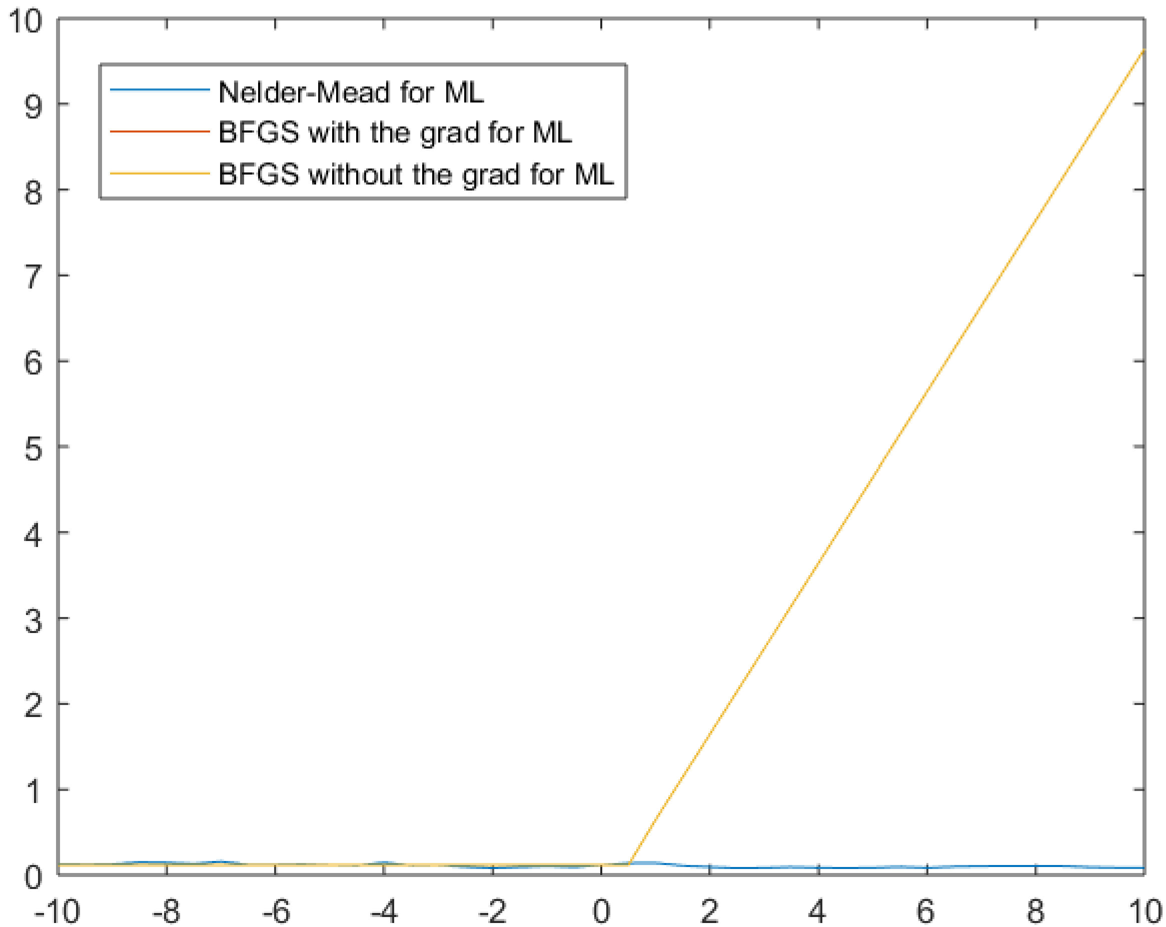

Figure 3: Comparison of 3

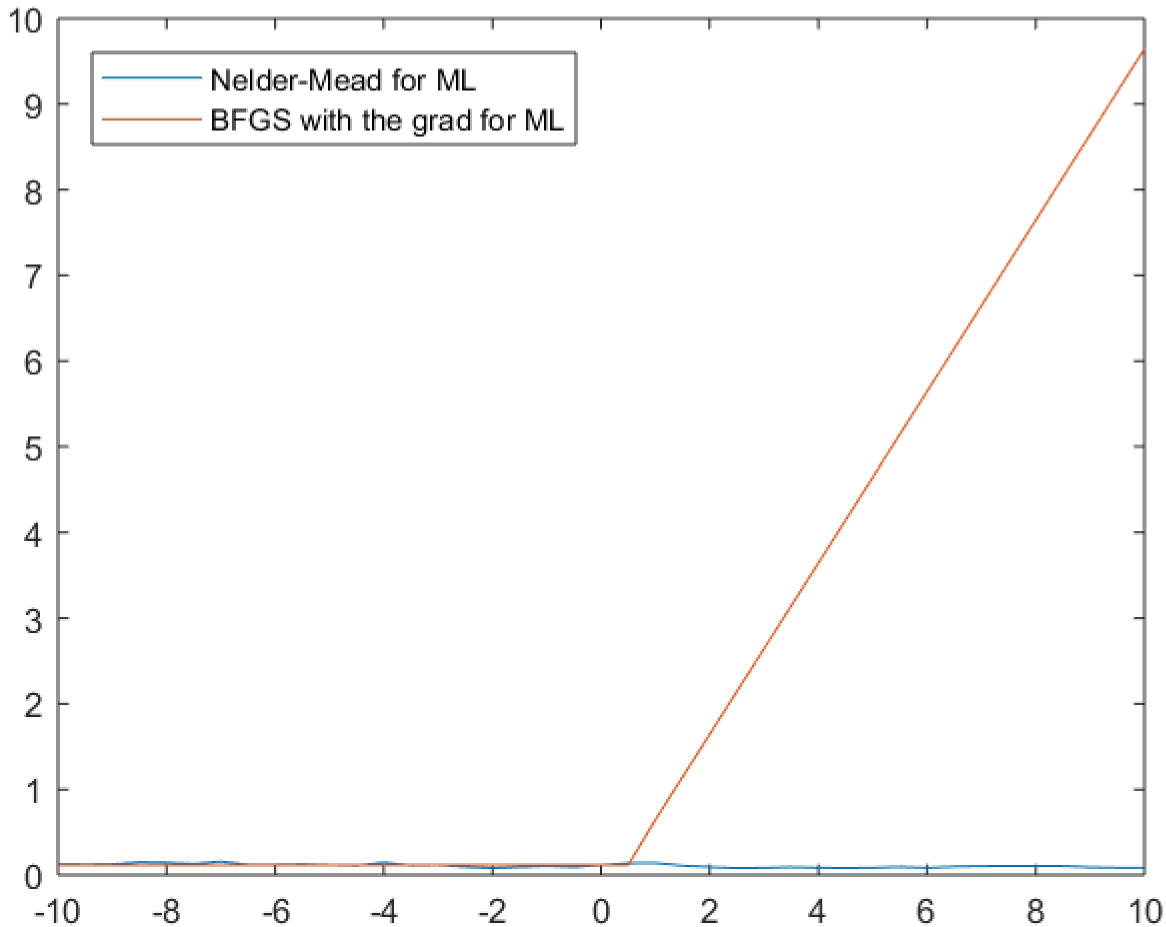

Figure 4: Comparison of 2

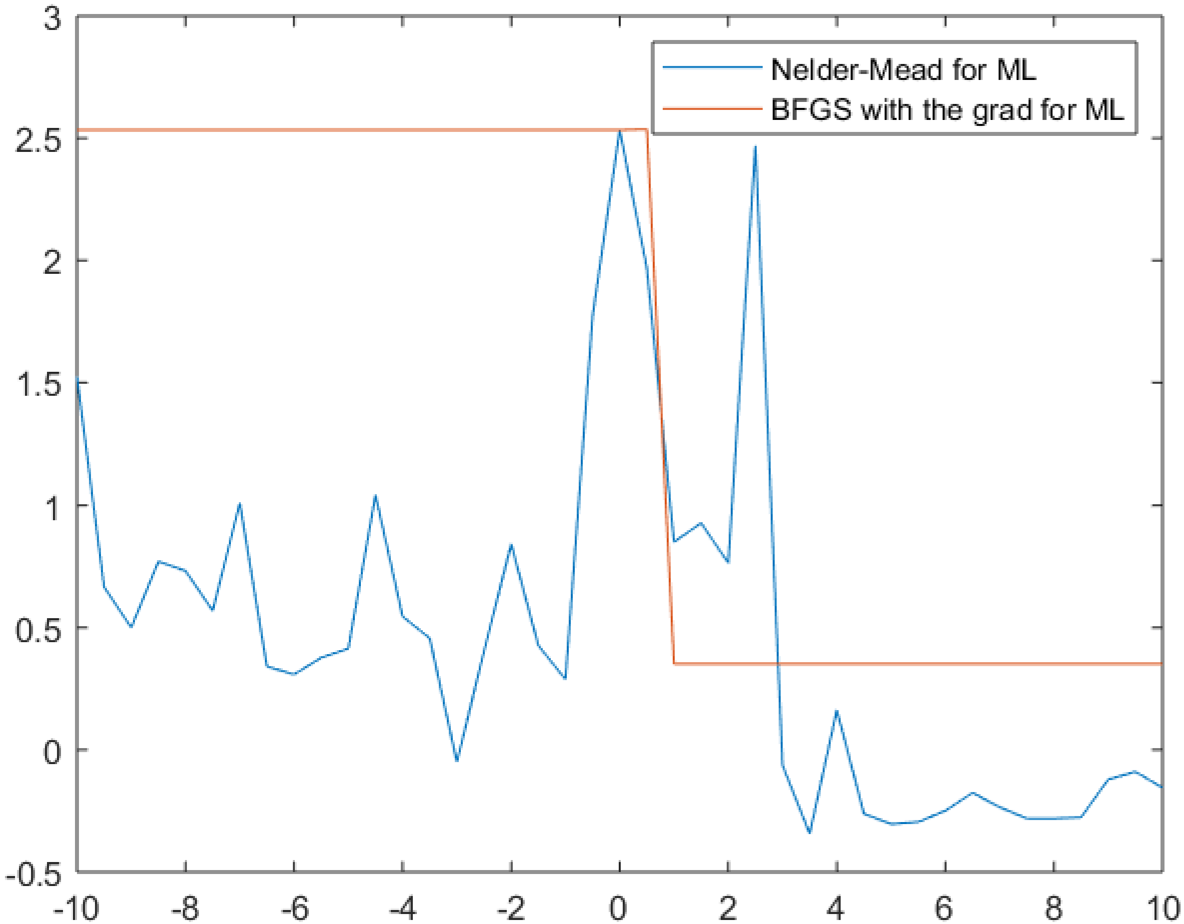

Figure 5: Comparison of 2 for $\hat{\beta}_0$

6: Code

The following is copied from HW3.m.

```
% Motoaki Takahashi
% HW3 for Econ 512 Empirical Method

clear
diary hw3.out

% x is the n by 6 matrix (explanatory), y is the n by 1 vector (explained)

%% Question 1

load('hw3.mat');
% we have X and y

% write the negative likelihood function as a function of a parameter
loglf_beta = @(beta) -loglf(X,y,beta)

% set the initial guess
beta = [log(mean(y)); zeros(5,1)];

[est_nm, valnm] = fminsearch(loglf_beta, beta)

%% Question 2

% I use the BFGS method.
% the function loglfgrad contains the objective function and the gradient

options = optimoptions('fminunc','Algorithm','quasi-newton',...
          'SpecifyObjectiveGradient',true, 'Display','iter', 'MaxFunctionEvaluations', 30000, 'MaxIterations', 10000);
[est_BFGS, valBFGS] = fminunc('loglfgrad', beta, options)
disp(est_BFGS);

% I calculate the BFGS outcome without the analytical gradient as well

[est_BFGS_wog, val_BFGS_wog] = fminunc(loglf_beta, beta)

% See the difference b/w the outcomes with and without the gradient
est_BFGS-est_BFGS_wog

%% Question 3
nls_beta=@(beta) nls(X,y,beta);
options1 = optimoptions(@lsqnonlin, 'MaxFunctionEvaluations', 30000, 'MaxIterations', 10000)
nls_com = lsqnonlin(nls_beta, beta, -Inf, +Inf, options1)

%% Question 4
nls_nm = fminsearch(nls_beta, beta)

%% Question 5

% See what happens if we move the initial guess for the 3rd element (the coefficient for the years married) from
% -10 to 10. The initial guess for beta_0 is log(mean(y)). The others are kept 0.

grid = [-10:0.5:10];

beta_mat = zeros(6, length(grid));

% the initial guess for beta_0 is log(mean(y))
beta_mat(1, [1:length(grid)]) = log(mean(y))*ones(1, length(grid));

beta_mat(3,[1:length(grid)]) = grid;
%
% %Nelder-Mead for ML
nm_mat = zeros(6, length(grid));
tic
for n=1:length(grid);
    nm_mat([1:6],n) =  fminsearch(loglf_beta, beta_mat([1:6],n));
end
toc
%
% %BFGS for ML
BFGS_mat = zeros(6, length(grid));
tic
for n=1:length(grid);
    BFGS_mat([1:6],n) =  fminunc('loglfgrad', beta_mat([1:6],n), options);
end
toc
% % BFGS for ML without the analytical gradient
BFGS_wog_mat =  zeros(6, length(grid));
tic
for n=1:length(grid);
    BFGS_wog_mat([1:6],n) =  fminunc(loglf_beta, beta_mat([1:6],n));
end
toc
```

```
% % %lsqnonlin
nls1_mat = zeros(6, length(grid));
tic
for n=1:length(grid);
    nls1_mat([1:6],n) =  lsqnonlin(nls_beta, beta_mat([1:6],n), -Inf, +Inf, options1);
end
toc
% %
% % %Nelder-Mead for NLS
nls2_mat = zeros(6, length(grid));
tic
for n=1:length(grid);
    nls2_mat([1:6],n) =  fminsearch(nls_beta, beta_mat([1:6],n));
end
toc

% Draw a figure
% I plot the estimates of beta_3 for each initial guess
plot(grid, nm_mat(3, [1:length(grid)]), grid, BFGS_mat(3, [1:length(grid)]), grid, BFGS_wog_mat(3, [1:length(grid)]), grid, nls1_mat(3, [1:length(grid)]), grid, nls2_mat(

legend('Nelder-Mead for ML', 'BFGS with the grad for ML', 'BFGS without the grad for ML', 'Built-in for NLS', 'Nelder-Mead for NLS')

% Since the nelder-mead for NLS is exceptionally bad, so plot the others
% separately
plot(grid, nm_mat(3, [1:length(grid)]), grid, BFGS_mat(3, [1:length(grid)]), grid, BFGS_wog_mat(3, [1:length(grid)]), grid, nls1_mat(3, [1:length(grid)]))

legend('Nelder-Mead for ML', 'BFGS with the grad for ML', 'BFGS without the grad for ML', 'Built-in for NLS')

% Plot the first three

plot(grid, nm_mat(3, [1:length(grid)]), grid, BFGS_mat(3, [1:length(grid)]), grid, BFGS_wog_mat(3, [1:length(grid)]))

legend('Nelder-Mead for ML', 'BFGS with the grad for ML', 'BFGS without the grad for ML')

% Plot the first two

plot(grid, nm_mat(3, [1:length(grid)]), grid, BFGS_mat(3, [1:length(grid)]))

legend('Nelder-Mead for ML', 'BFGS with the grad for ML')

diary off
```