Asha Bharadwaj, Caitlin Dutta, John Higgins, Alexis Smith
Econ 899
4 November, 2022

Problem 1.

Solution: We find the conditional log-likelihood, score of the log-likelihood, and the Hessian matrix evaluated at β^0, with $\beta_0^0 = -1$ and $\beta_j^0 = 0 \ \forall j \geq 1$ and include our computations below:

$$L(\beta^0) = -6924.3949$$

$$g(\beta^0) = \begin{bmatrix} -2597.54 \\ -554.361 \\ -1153.02 \\ -220.36 \\ -929.123 \\ -1210.18 \\ -2102.73 \\ -944.262 \\ -5033.67 \\ -4519.74 \\ -19342.5 \\ -19103.9 \\ -913.056 \\ -350.064 \\ -464.386 \\ -580.78 \\ -544.26 \end{bmatrix}$$

$$\mathrm{H}(\beta^0) = \begin{bmatrix}
-3215.6 & -878.3 & -1424.8 & -385.8 & -1301.4 & -1541.1 & -2611.9 & -1206.8 & -6286.9 & -5745.2 & -23719.5 & -23534.5 & -1399.3 & -662.4 & -678.7 & -672.4 & -581.4 \\
-878.3 & -878.3 & 0.0 & -10.0 & -403.4 & -420.0 & -684.4 & -331.0 & -1716.4 & -1651.0 & -6592.6 & -6547.2 & -388.5 & -163.4 & -188.7 & -211.4 & -169.1 \\
-1424.8 & 0.0 & -1424.8 & -164.5 & -558.2 & -674.0 & -1189.2 & -543.1 & -2788.9 & -2544.9 & -10487.9 & -10403.7 & -584.9 & -283.1 & -307.1 & -298.7 & -266.4 \\
-385.8 & -10.0 & -164.5 & -713.1 & -184.9 & -186.8 & -323.4 & -151.3 & -780.5 & -690.9 & -2727.1 & -2708.5 & 44.9 & -59.4 & -103.8 & -91.9 & -77.2 \\
-1301.4 & -403.4 & -558.2 & -184.9 & -1301.4 & -690.6 & -969.7 & -499.3 & -2547.7 & -2584.3 & -9523.6 & -9556.4 & -499.6 & -213.7 & -289.4 & -298.5 & -192.1 \\
-1541.1 & -420.0 & -674.0 & -186.8 & -690.6 & -802.7 & -1226.8 & -582.8 & -3013.6 & -2831.0 & -11394.8 & -11318.6 & -656.9 & -310.9 & -324.4 & -324.3 & -281.9 \\
-2611.9 & -684.4 & -1189.2 & -323.4 & -969.7 & -1226.8 & -2218.0 & -989.3 & -5111.0 & -4607.4 & -19165.8 & -18997.3 & -1223.2 & -543.6 & -548.8 & -538.4 & -476.1 \\
-1206.8 & -331.0 & -543.1 & -151.3 & -499.3 & -582.8 & -989.3 & -526.7 & -2362.2 & -2162.1 & -8842.7 & -8771.0 & -555.3 & -247.3 & -255.9 & -252.6 & -219.6 \\
-6286.9 & -1716.4 & -2788.9 & -780.5 & -2547.7 & -3013.6 & -5111.0 & -2362.2 & -12429.4 & -11233.2 & -46358.1 & -45992.1 & -2707.5 & -1293.9 & -1326.9 & -1315.3 & -1131.5 \\
-5745.2 & -1651.0 & -2544.9 & -690.9 & -2584.3 & -2831.0 & -4607.4 & -2162.1 & -11233.2 & -10804.4 & -42497.9 & -42187.5 & -2409.7 & -1166.9 & -1217.8 & -1210.5 & -1030.4 \\
-23719.5 & -6592.6 & -10487.9 & -2727.1 & -9523.6 & -11394.8 & -19165.8 & -8842.7 & -46358.1 & -42497.9 & -176268.1 & -174611.3 & -10022.4 & -4894.4 & -4995.0 & -4956.5 & -4260.5 \\
-23534.5 & -6547.2 & -10403.7 & -2708.5 & -9556.4 & -11318.6 & -18997.3 & -8771.0 & -45992.1 & -42187.5 & -174611.3 & -173634.8 & -9938.3 & -4836.8 & -4956.1 & -4931.3 & -4235.4 \\
-1399.3 & -388.5 & -584.9 & 44.9 & -499.6 & -656.9 & -1223.2 & -555.3 & -2707.5 & -2409.7 & -10022.4 & -9938.3 & -1399.3 & -292.0 & -304.9 & -304.4 & -267.2 \\
-662.4 & -163.4 & -283.1 & -59.4 & -213.7 & -310.9 & -543.6 & -247.3 & -1293.9 & -1166.9 & -4894.4 & -4836.8 & -292.0 & -662.4 & 0.0 & 0.0 & 0.0 \\
-678.7 & -188.7 & -307.1 & -103.8 & -289.4 & -324.4 & -548.8 & -255.9 & -1326.9 & -1217.8 & -4995.0 & -4956.1 & -304.9 & 0.0 & -678.7 & 0.0 & 0.0 \\
-672.4 & -211.4 & -298.7 & -91.9 & -298.5 & -324.3 & -538.4 & -252.6 & -1315.3 & -1210.5 & -4956.5 & -4931.3 & -304.4 & 0.0 & 0.0 & -672.4 & 0.0 \\
-581.4 & -169.1 & -266.4 & -77.2 & -192.1 & -281.9 & -476.1 & -219.6 & -1131.5 & -1030.4 & -4260.5 & -4235.4 & -267.2 & 0.0 & 0.0 & 0.0 & -581.4
\end{bmatrix}$$

(I'm so sorry for how terrible this looks). □

Problem 2.

Solution: We now find the numerical first and second derivatives of the log-likelihood. To do so, we use the ForwardDiff.jl library. We find that they are numerically equivalent to the above results. The maximum absolute difference between the analytic score function and the numerical score function is approximately $4.547 \times 10e-11$, whereas the maximum difference between the analytic Hessian and the numerical Hessian is approximately $7.101 \times 10e-9$. Since they are exceedingly similar, we refrained from plotting what is effectively the same table (especially with rounding) twice. All of the code to find each object and compare the analytic and numeric results is included in our submission. □

Problem 3.

Solution: We wrote a routine which implements the Newton algorithm and include the

estimated coefficient vector (rounded to four decimal places) below:

$$\begin{bmatrix} -6.0833 \\ 0.8718 \\ 0.5316 \\ 0.6037 \\ 0.1633 \\ 0.877 \\ -0.0614 \\ 0.227 \\ 1.0107 \\ 0.3347 \\ -0.287 \\ 0.194 \\ 0.7654 \\ 1.1542 \\ 0.7712 \\ 0.3779 \\ 0.2444 \end{bmatrix}$$

To determine the speed of our algorithm, we use the @belapsed command in BenchmarkTools.jl. This command runs the procedure a number of times and records the minimum time it takes for our procedure to complete. This avoids problems where initial function compilation causes inflated runtime estimates. We start from the initial guess and use tolerance level 1×10^{-12}. We find that our implementation takes a minimum of 0.1644 seconds to complete. □

Problem 4.

Solution: We now use BFGS and Nelder-Mead. We include our (rounded) results in the

following table (including the results of Newton's method for comparison):

Newton	BGFS	Nelder-Mead
−6.0833	−6.0833	−6.0833
0.8718	0.8718	0.8718
0.5316	0.5316	0.5316
0.6037	0.6037	0.6037
0.1633	0.1633	0.1633
0.877	0.877	0.877
−0.0614	−0.0614	−0.0614
0.227	0.227	0.227
1.0107	1.0107	1.0107
0.3347	0.3347	0.3347
−0.287	−0.287	−0.287
0.194	0.194	0.194
0.7654	0.7654	0.7654
1.1542	1.1542	1.1542
0.7712	0.7712	0.7712
0.3779	0.3779	0.3779
0.2444	0.2444	0.2444

We see (reassuringly) that they are all numerically identical.

For speed comparisons (with the same tolerance level of 1×10^{-12}), we found that when we did not pass the gradient or the Hessian matrix, BFGS takes 2.0030 seconds to find the minimizer (again, measured using @belapsed). However, when we passed the analytic gradient and Hessian matrix, BFGS takes only 0.3031 seconds. Nelder-Mead takes 4.5049 seconds to arrive at the minimizer. We note that Nelder-Mead did not initially converge to the true parameter in 1000 iterations. However, when we increased the max number of

iterations to 50000, we achieved convergence (albeit slowly).

Overall, we find that Newton's method is significantly faster than Nelder-Mead (which is derivative-free and does not benefit from any of the information which derivatives provide). It is also significantly faster than BFGS when a gradient and Hessian are not supplied. This is due to the fact that when analytic gradients/Hessians are not supplied, BFGS must numerically approximate these objects, which lengthens the computation time significantly. Finally, when we supply BFGS with the gradient and Hessian, we see a significant improvement in speed of BFGS making it very close to the speed of Newton's algorithm. □