```
title: Problem Set 1 (Econ899, Jean-Francois Houde)
author: Heejin Yoon
date: November 8, 2021

Error: syntax: extra token "Set" after end of expression
```

1 Data Import

```
using StatFiles, DataFrames, Optim

cd("/Users/Heejin/OneDrive - UW-Madison/3.Wisconsin/2021 Fall/Econ
899/github/Compuational-Economics/Second Half/Problem Sets/ps1");

include("ps1_model.jl");

data = DataFrame(load("Mortgage_performance_data.dta"));
data[!, :constant] .= 1.0;
y = select(data, :i_close_first_year);
X = select(data, [:constant, :i_large_loan, :i_medium_loan, :rate_spread, :i_refinance,
:age_r, :cltv, :dti, :cu, :first_mort_r, :score_0, :score_1, :i_FHA, :i_open_year2,
:i_open_year3, :i_open_year4, :i_open_year5]);
y = Float64.(Array(y));
X = Float64.(Array(X));
β_0 = [-1.0];
β_initial = append!(β_0, zeros(16));
```

2 Exercise 1.

Calculations of log-likelyhood estimator, Score, and Hessian are done in L1, g1, and H1, respectively.

2.1 Log-likelihood

```
L1 = loglikelihood(X, β_initial, y)

-6942.804936986123
```

2.2 Score

```
g1 = score(X, β_initial, y)

17-element Vector{Float64}:
  -2605.908251889911
   -556.3196848948576
  -1156.859426252986
   -222.81767101774338
   -933.0399793180802
  -1215.1317434743728
  -2109.626210065904
   -948.0740374412838
```

```
  -5049.8756176514335
  -4534.7904704059665
 -19401.89853109084
 -19164.65945479787
   -918.8553971099461
   -351.7530628092251
   -466.6888493111679
   -582.4690752990939
   -546.4113143620536
```

2.3 Hessian

```
H1 = hessian(X, β_initial, y)
```

```
17×17 Matrix{Float64}:
  -3224.63    -880.428    -1428.39   …   -681.85    -674.379    -582.954
   -880.428   -880.428        0.0        -189.337   -211.554    -170.069
  -1428.39       0.0      -1428.39       -308.877   -299.44     -266.606
   -387.591    -10.0764    -165.227       -104.651    -92.3703    -77.2665
  -1305.7     -404.234     -560.344       -290.986   -299.44     -192.876
  -1546.77    -421.328     -675.977   …   -326.323   -325.511    -282.993
  -2619.43    -686.268    -1192.14        -551.432   -540.134    -477.388
  -1210.69    -331.979     -544.713       -257.316   -253.334    -220.353
  -6304.56   -1720.75     -2796.02       -1333.23   -1318.87    -1134.65
  -5761.32   -1655.08     -2551.24       -1223.71   -1213.88    -1033.59
 -23783.2    -6607.99    -10512.7    …   -5017.54   -4970.07    -4271.64
 -23599.2    -6563.05    -10428.5        -4978.92   -4944.78    -4247.05
  -1404.99    -390.078     -586.887       -306.911   -305.732    -268.375
   -664.352   -163.778     -283.711          0.0        0.0         0.0
   -681.85    -189.337     -308.877       -681.85       0.0         0.0
   -674.379   -211.554     -299.44    …      0.0     -674.379       0.0
   -582.954   -170.069     -266.606          0.0        0.0      -582.954
```

3 Exercise 2.

Numerical calculations of score and Hessian are done in g2 and H2, respectively.

3.1 Score

```
g2 = score_numerical(X, β_initial, y)
```

```
17-element Vector{Float64}:
  -2628.662514325697
   -563.3910404867493
  -1167.1545507851988
   -222.66704036155716
   -941.822690947447
  -1215.125848830212
  -2109.0909285703674
   -948.9804142503999
  -5036.845323047601
  -4528.087629296351
 -19401.782083150465
```

```
-19164.96330522932
  -929.221641854383
  -349.72345019923523
  -467.20288082724437
  -595.618985244073
  -557.8840500675142
```

3.2 Hessian

```
H2 = hessian_numerical(X, β_initial, y)
```

```
17×17 Matrix{Float64}:
 -1.78034e10  -4.84306e9  -7.7307e9   …  -4.59295e9  -4.54747e7   0.0
 -4.84306e9   -4.84306e9   0.0           -1.59162e9  -2.27374e7   0.0
 -7.7307e9     0.0        -7.70797e9     -2.0691e9    0.0         0.0
 -2.27374e8   -6.59384e8  -3.18323e8     -7.27596e8   5.91172e8  -2.72848e8
 -7.18501e9   -2.22826e9  -2.95586e9     -2.47837e9  -2.27374e7   0.0
  1.20508e9    3.86535e8   6.59384e8  …   1.36424e8   5.45697e8   7.04858e8
 -6.82121e8    2.04636e8   9.09495e7     -1.09139e9   1.29603e9   4.77485e8
  9.09495e8    2.27374e8   3.41061e8     -4.54747e8  -2.27374e7   0.0
  1.68257e9    4.3201e8    7.27596e8     -1.13687e8  -2.27374e7   0.0
 -1.59162e9    1.36424e8  -7.04858e8     -5.68434e8  -2.27374e7   0.0
 -3.41061e8    2.72848e8  -8.86757e8  …  -2.27374e8  -9.09495e7  -1.81899e8
  1.13687e8    9.09495e7  -4.54747e7      4.54747e8  -2.50111e8  -3.18323e8
 -8.34461e9   -2.41016e9  -3.52429e9     -2.66027e9  -2.27374e7   0.0
 -1.02773e10  -2.56932e9  -4.34284e9      0.0         0.0         0.0
 -4.59295e9   -1.59162e9  -2.0691e9      -4.57021e9   0.0         0.0
 -4.54747e7   -2.27374e7   0.0        …   0.0        -4.54747e7   0.0
  0.0          0.0         0.0            0.0         0.0         0.0
```

4 Exercise 3.

The estimated coefficients using a Newton algorithm are calculated in β_{newton}.

```
speed_newton = @elapsed β_newton = Newton_method(X, β_initial, y);
β_newton
```

```
Iteration #1 / Difference: 2.671091227045415
Iteration #2 / Difference: 1.8934320388091663
Iteration #3 / Difference: 1.2100194564583564
Iteration #4 / Difference: 0.27140447074170737
Iteration #5 / Difference: 0.010475923172530877
Iteration #6 / Difference: 1.6690567110622112e-5
Iteration #7 / Difference: 5.1731952055433794e-11
Iteration #8 / Difference: 7.638334409421077e-14
17-element Vector{Float64}:
 -6.056439806846095
  0.8675934654320824
  0.5273597052861058
  0.5955822205769957
  0.16338862333997145
  0.8712242569232365
 -0.05662401835859635
  0.215082747334844
  1.0079202312043756
```

```
  0.3355954315371512
 -0.28418643217308703
  0.18943330726652086
  0.7585839588514371
  1.152719757818077
  0.7701571783836607
  0.3793401435982599
  0.24021247150953362
```

5 Exercise 4.

Calculations of estimates using BFGS and Simplex methods are calculated in β_{bfgs} and $\beta_{simplex}$, respectively.

```
speed_bfgs = @elapsed res_bfgs = optimize(β -> -loglikelihood(X, β, y), β_initial,
BFGS());
β_bfgs = res_bfgs.minimizer;
speed_simplex = @elapsed res_simplex = optimize(β -> -loglikelihood(X, β, y), β_newton,
NelderMead());
β_simplex = res_simplex.minimizer;
```

5.1 β_{bfgs}

```
β_bfgs
```

```
17-element Vector{Float64}:
 -6.056439845646469
  0.8675934697885819
  0.5273597128986782
  0.5955822194248732
  0.16338862348493102
  0.8712242709854191
 -0.05662402798952915
  0.2150827523589919
  1.0079202368972364
  0.33559543446714907
 -0.28418643590400605
  0.1894333113950758
  0.7585839643854538
  1.1527197678135175
  0.7701571941814083
  0.3793401610050737
  0.24021247059493897
```

5.2 $\beta_{simplex}$

```
β_simplex
```

```
17-element Vector{Float64}:
 -6.056439806846095
  0.8675934654320824
  0.5273597052861058
  0.5955822205769957
```

```
  0.16338862333997145
  0.8712242569232365
 -0.05662401835859635
  0.215082747334844
  1.0079202312043756
  0.3355954315371512
 -0.28418643217308703
  0.18943330726652086
  0.7585839588514371
  1.152719757818077
  0.7701571783836607
  0.3793401435982599
  0.24021247150953362
```

Finally, computing speed comparison is presented below.

5.3 Newton Algorithm

```
speed_newton
```

```
0.6832543
```

5.4 BFGS

```
speed_bfgs
```

```
45.2424724
```

5.5 Simplex

```
speed_simplex
```

```
12.7150487
```