Assignment 3

Arthur Yu, Hristina Hristova

4/20/2017

Assignment 3

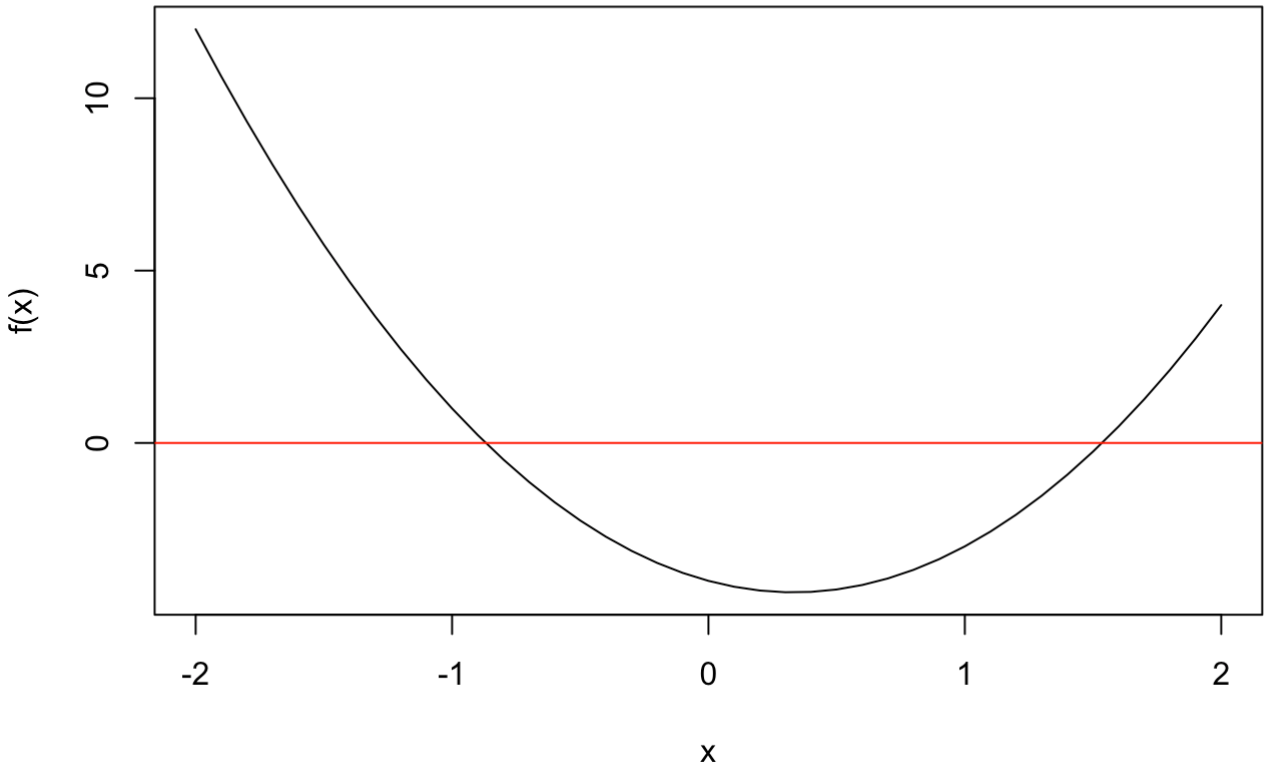
1. Derive the analytical version of the Newton-Raphson algorithm (including $f(x)$ and $f'(x)$ for each expression below (sextely). For each expression below, implement the Newton-Raphson algorithm, and find at least one root for each non-linear function below. Also plot $f(x)$ around the solution(s). Prepare a data table with values for $x, f(x), f'(x)$, and err for each iteration.

- $3x^2 - 2x - 4 = 0$

```
tol <- 10 ^ (-5)
err <- 10 ^ 6
x0 <- 0
x <- NULL
f <- function(x){3 * x ^ 2 - 2 * x - 4}
fp <- function(x){6 * x - 2}
k <- 0
dta <- NULL
while (err > tol){
  x <- x0 - f(x0) / fp(x0)
  err <- abs(x - x0)
  dta <- rbind(dta, data.frame(k = k, x = x0, f = f(x0), err = err))
  x0 <- x
  k <- k + 1
}
dta
```

```
##   k          x             f          err
## 1 0  0.0000000 -4.000000e+00 2.000000e+00
## 2 1 -2.0000000  1.200000e+01 8.571429e-01
## 3 2 -1.1428571  2.204082e+00 2.488479e-01
## 4 3 -0.8940092  1.857759e-01 2.522739e-02
## 5 4 -0.8687818  1.909263e-03 2.647088e-04
## 6 5 -0.8685171  2.102123e-07 2.915120e-08
```

```
x <- seq(-2, 2, by = 0.1)
plot(x, f(x), type = "l")
abline(h = 0, col = "red")
```

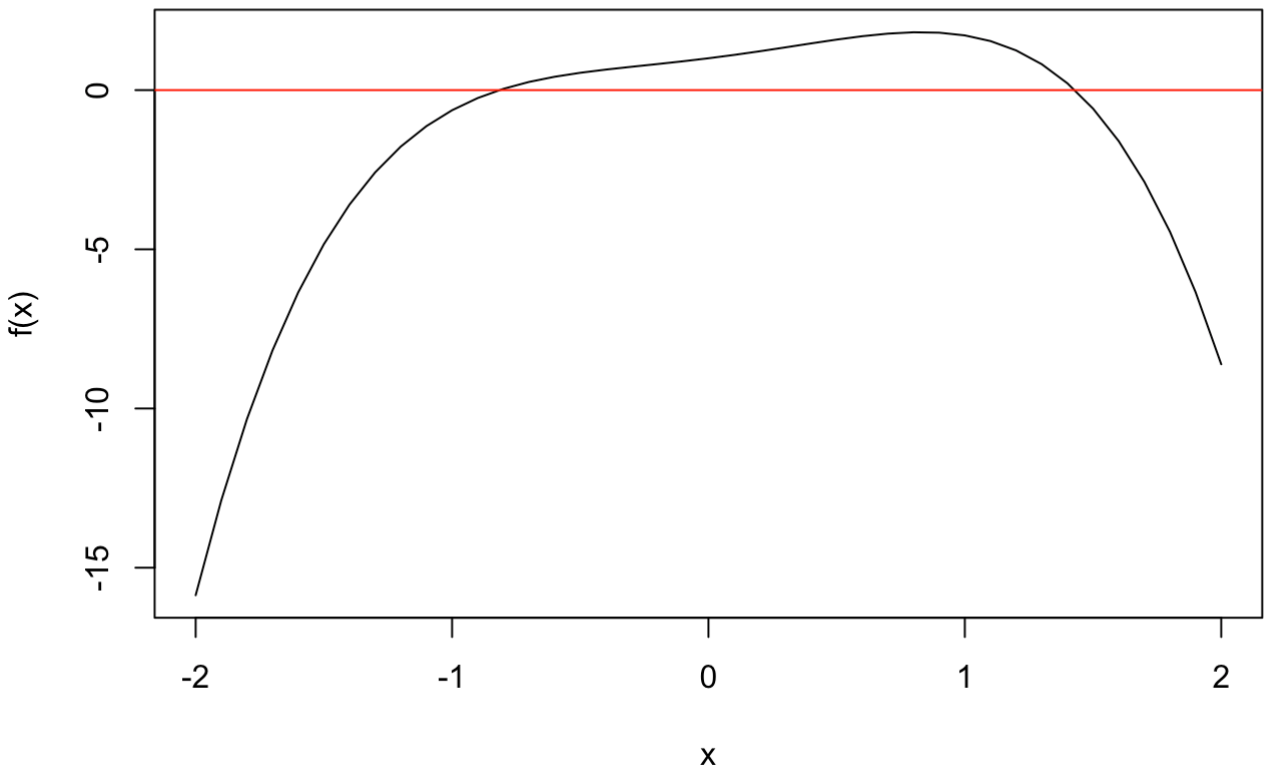


- $e^x - x^4 = 0$

```
err <- 10 ^ 5
tol <- 10 ^ (-5)
k <- 0
x <- NULL
X0 <- 0
dta <- NULL
f <- function(x){exp(x) - x ^ 4}
fp <- function(x){exp(x) - 4 * x ^ 3}
while (err > tol){
  x <- x0 - f(x0) / fp(x0)
  err <- abs(x - x0)
  dta <- rbind(dta, data.frame(k = k, x = x0, f = f(x0), fp = fp(x0), err = err))
  x0 <- x
  k <- k + 1
}
dta
```

```
##   k          x             f       fp          err
## 1 0 -0.8685171 -1.494283e-01 3.040139 4.915180e-02
## 2 1 -0.8193653 -1.001227e-02 2.641066 3.790996e-03
## 3 2 -0.8155743 -5.454202e-05 2.612339 2.087861e-05
## 4 3 -0.8155534 -1.643277e-09 2.612182 6.290820e-10
```

```
x <- seq(-2 , 2, by = 0.1)
plot(x, f(x), type = "l")
abline(h = 0, col = "red")
```

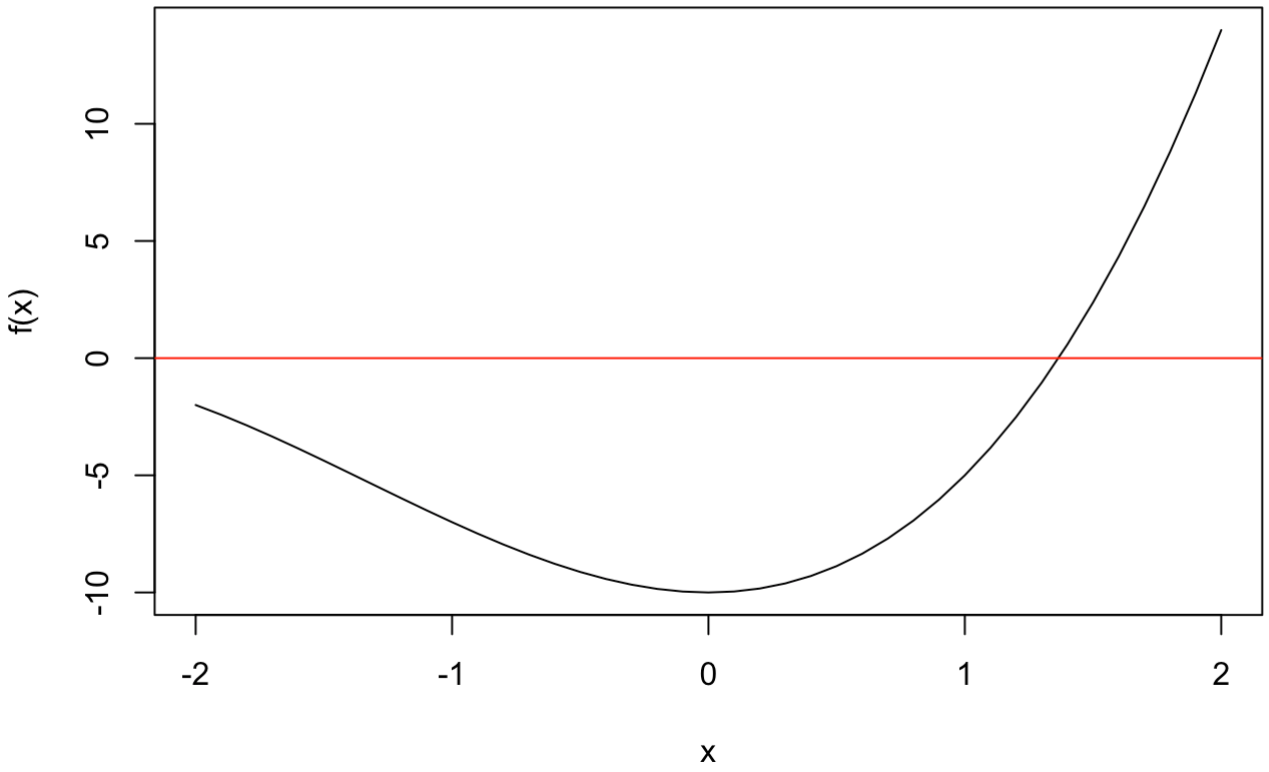


- $x^3 + 4x^2 - 10 = 0$

```
err <- 10 ^ 5
tol <- 10 ^ (-5)
k <- 0
x <- NULL
x0 <- 1
f <- function(x){x ^ 3 + 4 * x ^ 2 - 10}
fp <- function(x){3 * x ^ 2 + 8 * x}
dta <- NULL
while (err > tol){
  x <- x0 - f(x0) / fp(x0)
  err <- abs(x - x0)
  dta <- rbind(dta, data.frame(k = k, x = x0, f = f(x0), fp = fp(x0), err = err))
  x0 <- x
  k <- k + 1
}
dta
```

```
##   k        x             f       fp          err
## 1 0 1.000000 -5.0000000000 11.00000 4.545455e-01
## 2 1 1.454545  1.5401953418 17.98347 8.564505e-02
## 3 2 1.368900  0.0607196886 16.57287 3.663801e-03
## 4 3 1.365237  0.0001087706 16.51351 6.586767e-06
```

```
x <- seq(-2, 2, by = 0.1)
plot(x, f(x), type = "l")
abline(h = 0, col = "red")
```



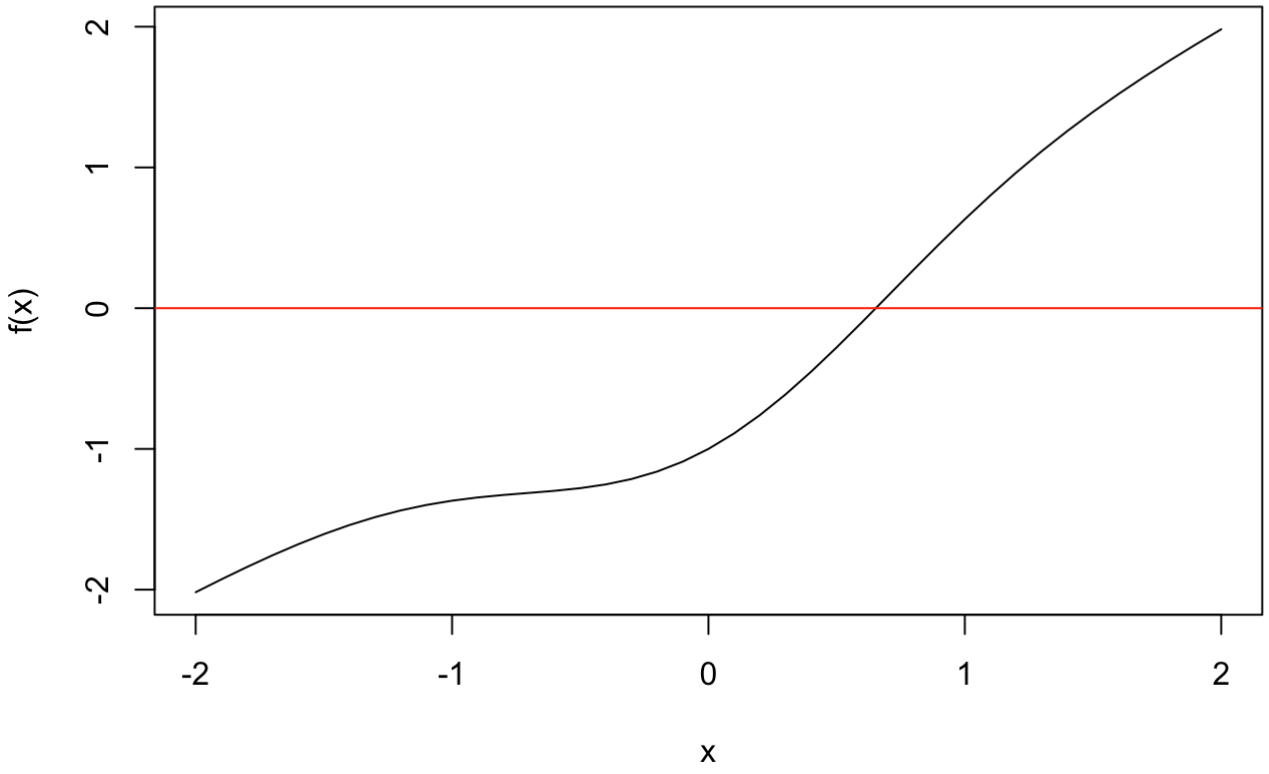
- $x - e^{-x^2} = 0$

```
err = 10 ^ 5
tol = 10 ^ (-5)
x <- NULL
x0 <- 0
k <- 0
dta <- NULL
f <- function(x){x - exp(- x ^ 2)}
fp <- function(x){1 + 2 * exp(- x ^ 2) * x}
while (err > tol){
  x <- x0 - f(x0) / fp(x0)
  err <- abs(x - x0)
  dta <- rbind(dta, data.frame(k = k, x = x0, f = f(x0), fp = fp(x0), err = err))
  k <- k + 1
  x0 <- x
}

dta
```

```
##   k         x             f       fp          err
## 1 0 0.0000000 -1.000000e+00 1.000000 1.000000e+00
## 2 1 1.0000000  6.321206e-01 1.735759 3.641753e-01
## 3 2 0.6358247 -3.163720e-02 1.848777 1.711250e-02
## 4 3 0.6529372  3.432590e-05 1.852609 1.852841e-05
## 5 4 0.6529186  3.303036e-11 1.852606 1.782918e-11
```

```
x <- seq(-2, 2, by = 0.1)
plot(x, f(x), type = "l")
abline(h = 0, col = "red")
```



2. Write a R code for $f(x)$ and $Df(x)$. Run the Newthon-Raphson method and search for the solution. Prepare a table with values for $x, f(x)$, $Df^{-1}(x)$, and err for each iteration. Explain what is happening during the iterations. Is the algorithm converging? Why or why not? Choose your initial values carefully.

$$f_1(x_1, x_2, x_3) = x_1 + x_2 + x_3^2 - 12$$
$$f_2(x_1, x_2, x_3) = x_1^2 - x_2 + x_3 - 2$$
$$f_3(x_1, x_2, x_3) = 2x_1^2 - x_2^2 + x_3 - 1$$

For the equations, we have 3 variables with up to the second power. We may have up to $2^3 = 8$ sets of solutions, which can be expressed as $(-,-,-), (+,-,-), (-,+,-), (-,-,+), (+,+,-), (+,-,+), (-,+,+), (+,+,+)$, let's use 3.0 as the initial value for positive solution, -3.0 for negative solution.

The $Df(x)$ is:

$$\begin{bmatrix} 1 & 1 & 2x_3 \\ 2x_1 & -1 & 1 \\ 4x_1 & -2x_2 & 1 \end{bmatrix}$$

```
err <- 10 ^ 5
tol <- 10 ^ (-5)
k <- 0
x <- NULL
f <- function(x){matrix(c(x[1] + x[2] + x[3] ^ 2 - 12,
                        x[1] ^ 2 - x[2] + x[3] - 2,
                        2 * x[1] ^ 2 - x[2] ^ 2 + x[3] - 1))}
Df <- function(x){matrix(c(1, 1, 2 * x[3],
                           2 * x[1], - 1, 1,
                           4 * x[1], - 2 * x[2], 1), byrow = T, nrow = 3)}
dta <- NULL
iter <- function(xi){
while(err > tol){
  x <- xi - solve(Df(xi)) %*% f(xi)
  err <- max(abs(x - xi))
  dta <- rbind(dta, data.frame(k = rep(k, 3), x = xi, f = f(xi), invDf = solve(Df(xi)),
err = err))
  xi <- x
  k <- k + 1
}
  dta
}
x1 <- c(-1, -1, -1)
x2 <- c(1, -1, -1)
x3 <- c(-1, 1, -1)
x4 <- c(-1, -1, 1)
x5 <- c(1, 1, -1)
x6 <- c(1, -1, 1)
x7 <- c(-1, 1, 1)
x8 <- c(1, 1, 1)
iter(x1)
```

```
##    k          x             f      invDf.1     invDf.2      invDf.3
## 1  0  -1.000000 -1.300000e+01 -0.272727273 -0.45454545 -0.090909091
## 2  0  -1.000000 -1.000000e+00 -0.181818182 -0.63636364  0.272727273
## 3  0  -1.000000 -1.000000e+00 -0.727272727 -0.54545455  0.090909091
## 4  1  -5.090909  9.819008e+01 -0.004061407 -0.07861184 -0.010000668
## 5  1  -3.727273  1.673554e+01 -0.004891156 -0.21295189  0.106235755
## 6  1 -10.909091  2.603306e+01 -0.046243659 -0.01336334  0.004410775
## 7  2  -3.116160  2.161771e+01 -0.011211168 -0.11845720 -0.021897787
## 8  2  -2.448787  3.899634e+00 -0.011847514 -0.29474207  0.146420546
## 9  2  -6.259606  6.164743e+00 -0.081719102 -0.03300522  0.009946534
## 10 3  -2.276866  3.363448e+00 -0.021452368 -0.15545036 -0.034430332
## 11 3  -1.945929  7.044141e-01 -0.019969574 -0.34912686  0.172370768
## 12 3  -4.425635  1.155963e+00 -0.117657915 -0.05700620  0.015584254
## 13 4  -2.055411  1.746216e-01 -0.026223628 -0.17075218 -0.039443691
## 14 4  -1.832087  4.904255e-02 -0.023112482 -0.36489454  0.179636093
## 15 4  -4.007757  8.512502e-02 -0.130913128 -0.06682624  0.017490131
## 16 5  -2.039100  6.075603e-04 -0.026598403 -0.17204895 -0.039839701
## 17 5  -1.825447  2.660480e-04 -0.023323165 -0.36587582  0.180078418
## 18 5  -3.983109  4.880095e-04 -0.131796752 -0.06752575  0.017604179
## 19 6  -2.039018  8.001047e-09 -0.026600076 -0.17205564 -0.039841588
## 20 6  -1.825424  6.621983e-09 -0.023323939 -0.36587938  0.180079987
## 21 6  -3.983019  1.268555e-08 -0.131800019 -0.06752855  0.017604535
##             err
## 1  9.909091e+00
## 2  9.909091e+00
## 3  9.909091e+00
## 4  4.649485e+00
## 5  4.649485e+00
## 6  4.649485e+00
## 7  1.833971e+00
## 8  1.833971e+00
## 9  1.833971e+00
## 10 4.178775e-01
## 11 4.178775e-01
## 12 4.178775e-01
## 13 2.464874e-02
## 14 2.464874e-02
## 15 2.464874e-02
## 16 8.944857e-05
## 17 8.944857e-05
## 18 8.944857e-05
## 19 1.857591e-09
## 20 1.857591e-09
## 21 1.857591e-09
```

```
iter(x2)
```

```
##    k         x             f     invDf.1     invDf.2    invDf.3
## 1  0  1.000000 -1.100000e+01  0.17647059  0.29411765 0.05882353
## 2  0 -1.000000 -1.000000e+00 -0.11764706 -0.52941176 0.29411765
## 3  0 -1.000000 -1.000000e+00 -0.47058824 -0.11764706 0.17647059
## 4  1  3.294118  2.619031e+01  0.01065802  0.11064016 0.01976391
## 5  1 -2.529412  5.262976e+00 -0.01158931 -0.28535629 0.14355769
## 6  1 -6.117647  8.186851e+00 -0.08180688 -0.01427968 0.01334840
## 7  2  2.270880  4.445435e+00  0.02269603  0.14879242 0.03319468
## 8  2 -1.899346  1.047015e+00 -0.02148084 -0.34921582 0.17697273
## 9  2 -4.009227  1.697048e+00 -0.12456077 -0.02499527 0.02621046
## 10 3  1.957865  2.866712e-01  0.02993396  0.16820393 0.03976585
## 11 3 -1.738552  9.797817e-02 -0.02618062 -0.37047198 0.18857899
## 12 3 -3.473810  1.701014e-01 -0.14339394 -0.02911328 0.03286663
## 13 4  1.926040  1.472158e-03  0.03072662  0.17061466 0.04050438
## 14 4 -1.726826  1.012878e-03 -0.02657626 -0.37210391 0.18950155
## 15 4 -3.435442  1.888260e-03 -0.14493765 -0.02932509 0.03347546
## 16 5  1.925745  3.235088e-08  0.03073266  0.17063852 0.04051091
## 17 5 -1.726768  8.674738e-08 -0.02657810 -0.37211170 0.18950622
## 18 5 -3.435262  1.701085e-07 -0.14494463 -0.02932428 0.03347884
##             err
## 1  5.117647e+00
## 2  5.117647e+00
## 3  5.117647e+00
## 4  2.108420e+00
## 5  2.108420e+00
## 6  2.108420e+00
## 7  5.354168e-01
## 8  5.354168e-01
## 9  5.354168e-01
## 10 3.836871e-02
## 11 3.836871e-02
## 12 3.836871e-02
## 13 2.945291e-04
## 14 2.945291e-04
## 15 2.945291e-04
## 16 2.268792e-08
## 17 2.268792e-08
## 18 2.268792e-08
```

```
iter(x3)
```

```
##     k           x             f      invDf.1       invDf.2       invDf.3
## 1   0  -1.0000000 -1.100000e+01 -1.000000000 -3.000000e+00  1.0000000000
## 2   0   1.0000000 -3.000000e+00  2.000000000  7.000000e+00 -3.0000000000
## 3   0  -1.0000000 -1.000000e+00  0.000000000  2.000000e+00 -1.0000000000
## 4   1 -20.0000000  2.500000e+01  0.003159003 -2.562303e-02  0.0003510004
## 5   1  41.0000000  3.610000e+02 -0.001560002  2.499902e-02 -0.0125190125
## 6   1   4.0000000 -8.780000e+02  0.124800125  7.800008e-05  0.0015210015
## 7   2 -10.5208845  3.285960e+00  0.011075037 -4.989856e-02  0.0014501876
## 8   2  21.0226590  8.985363e+01 -0.005677587  4.994367e-02 -0.0251067493
## 9   2   2.1872782 -2.193869e+02  0.227360780 -1.031271e-05  0.0054077625
## 10  3  -5.7555577  1.937939e-01  0.017218509 -9.558969e-02  0.0051064752
## 11  3  11.0456032  2.270834e+01 -0.009397483  9.958387e-02 -0.0502001262
## 12  3   2.6274985 -5.412496e+01  0.188806762 -7.600728e-04  0.0085810993
## 13  4  -3.3118236  1.981337e-01  0.026698932 -1.811994e-01  0.0171280008
## 14  4   6.0689601  5.971836e+00 -0.015877678  1.975414e-01 -0.0999692778
## 15  4   3.0726206 -1.282330e+01  0.160966628 -2.659303e-03  0.0134805576
## 16  5  -2.0153830  2.460296e-02  0.045041769 -3.449276e-01  0.0540051252
## 17  5   3.6104844  1.680758e+00 -0.029184013  3.842363e-01 -0.1957383271
## 18  5   3.2294739 -2.682586e+00  0.152368819 -6.085942e-03  0.0219436984
## 19  6  -1.2918780  4.270118e-03  0.077084598 -6.587204e-01  0.1507606072
## 20  6   2.4403091  5.234596e-01 -0.051323726  6.962738e-01 -0.3580688826
## 21  6   3.2948200 -3.223911e-01  0.147844058 -5.698855e-03  0.0314597268
## 22  7  -0.8987897  1.561037e-04  0.121881815 -1.123905e+00  0.3177025701
## 23  7   1.9606188  1.545183e-01 -0.074999803  1.033914e+00 -0.5378184853
## 24  7   3.3073142  7.893395e-02  0.144092448  1.360487e-02  0.0332771403
## 25  8  -0.7502223  2.257572e-05  0.150075813 -1.416091e+00  0.4248213511
## 26  8   1.8433240  2.207229e-02 -0.083814644  1.163072e+00 -0.6094661201
## 27  8   3.3025628  3.038649e-02  0.141365796  3.830640e-02  0.0279547705
## 28  9  -0.7318781  2.883710e-06  0.153857087 -1.453383e+00  0.4376601258
## 29  9   1.8361737  3.365079e-04 -0.084273950  1.170281e+00 -0.6139275616
## 30  9   3.3008646  6.218898e-04  0.140935326  4.288295e-02  0.0267001916
## 31 10  -0.7316617  9.885621e-10  0.153897832 -1.453758e+00  0.4377762623
## 32 10   1.8361620  4.685267e-08 -0.084272079  1.170262e+00 -0.6139256645
## 33 10   3.3008332  9.356679e-08  0.140930213  4.294319e-02  0.0266825665
##             err
## 1  4.000000e+01
## 2  4.000000e+01
## 3  4.000000e+01
## 4  1.997734e+01
## 5  1.997734e+01
## 6  1.997734e+01
## 7  9.977056e+00
## 8  9.977056e+00
## 9  9.977056e+00
## 10 4.976643e+00
## 11 4.976643e+00
## 12 4.976643e+00
## 13 2.458476e+00
## 14 2.458476e+00
## 15 2.458476e+00
## 16 1.170175e+00
## 17 1.170175e+00
## 18 1.170175e+00
```

```
## 19 4.796902e-01
## 20 4.796902e-01
## 21 4.796902e-01
## 22 1.485675e-01
## 23 1.485675e-01
## 24 1.485675e-01
## 25 1.834415e-02
## 26 1.834415e-02
## 27 1.834415e-02
## 28 2.164548e-04
## 29 2.164548e-04
## 30 2.164548e-04
## 31 2.699900e-08
## 32 2.699900e-08
## 33 2.699900e-08
```

```
iter(x4)
```

```
##     k           x             f      invDf.1       invDf.2      invDf.3
## 1   0  -1.00000000 -1.300000e+01  0.142857143 -1.428571e-01 -0.142857143
## 2   0  -1.00000000  1.000000e+00  0.095238095 -4.285714e-01  0.238095238
## 3   0   1.00000000  1.000000e+00  0.380952381  2.857143e-01 -0.047619048
## 4   1   1.14285714  2.222449e+01 -0.004464692 -3.374032e-01  0.388428246
## 5   1   0.42857143  4.591837e+00  0.071435080 -1.601549e+00  0.785148064
## 6   1   5.71428571  7.142857e+00  0.081640091  1.696583e-01 -0.102687927
## 7   2   0.01689554  3.459479e+00  0.410784230 -2.377019e+01 20.603605386
## 8   2   0.58681419  1.267790e+00  0.079945713  1.133337e+00 -1.749609447
## 9   2   3.85431826  2.510538e+00  0.066064868  2.936558e+00 -2.445827600
## 10  3 -22.99474385  4.791095e+00  0.002231901 -3.225551e-02  0.005279969
## 11  3   3.26587239  5.295355e+02 -0.018555445  4.489387e-01 -0.224671076
## 12  3   6.04317516  1.051894e+03  0.084088539 -3.447552e-02  0.018151973
## 13  4 -11.47896595  1.539613e+00  0.007252092 -9.606669e-02  0.026412323
## 14  4   1.95588216  1.326131e+02 -0.057179434  1.100876e+00 -0.551683490
## 15  4   4.80236368  2.635102e+02  0.109313602 -1.046162e-01  0.054688816
## 16  5  -5.71034255  4.982731e-01  0.026337027 -3.463020e-01  0.130523818
## 17  5   1.42747511  3.327702e+01 -0.162153619  2.671234e+00 -1.342716047
## 18  5   4.09647904  6.627482e+01  0.138633273 -2.837720e-01  0.147955380
## 19  6  -2.85001146  1.863724e-01  0.046923272 -5.206463e-01  0.176720262
## 20  6   1.60584180  8.181494e+00 -0.120932183  1.793971e+00 -0.907593451
## 21  6   3.66477039  1.633117e+01  0.146531542 -1.737250e-01  0.099716095
## 22  7  -1.48514136  5.497157e-02  0.081913332 -8.148619e-01  0.252885611
## 23  7   1.77308471  1.862870e+00 -0.095557567  1.343340e+00 -0.687755397
## 24  7   3.43031022  3.697771e+00  0.147748187 -7.703060e-02  0.063386364
## 25  8  -0.90677516  9.803408e-03  0.127954757 -1.224172e+00  0.371661106
## 26  8   1.81903179  3.345075e-01 -0.087963153  1.220629e+00 -0.634566252
## 27  8   3.33129806  6.669038e-01  0.144089238  5.317385e-04  0.039459865
## 28  9  -0.74639709  7.787655e-04  0.151310660 -1.430830e+00  0.431152744
## 29  9   1.83477919  2.572113e-02 -0.084611626  1.174701e+00 -0.615690718
## 30  9   3.30339169  5.119427e-02  0.141264048  3.876744e-02  0.027931591
## 31 10  -0.73178493  6.436832e-06  0.153875833 -1.453563e+00  0.437719977
## 32 10   1.83615027  2.135152e-04 -0.084274964  1.170299e+00 -0.613940588
## 33 10   3.30085460  4.251505e-04  0.140933068  4.290769e-02  0.026693180
## 34 11  -0.73166166  4.586962e-10  0.153897835 -1.453758e+00  0.437776270
## 35 11   1.83616196  1.519563e-08 -0.084272079  1.170262e+00 -0.613925662
## 36 11   3.30083318  3.025476e-08  0.140930212  4.294319e-02  0.026682565
##             err
## 1  4.714286e+00
## 2  4.714286e+00
## 3  4.714286e+00
## 4  1.859967e+00
## 5  1.859967e+00
## 6  1.859967e+00
## 7  2.301164e+01
## 8  2.301164e+01
## 9  2.301164e+01
## 10 1.151578e+01
## 11 1.151578e+01
## 12 1.151578e+01
## 13 5.768623e+00
## 14 5.768623e+00
## 15 5.768623e+00
```

```
## 16 2.860331e+00
## 17 2.860331e+00
## 18 2.860331e+00
## 19 1.364870e+00
## 20 1.364870e+00
## 21 1.364870e+00
## 22 5.783662e-01
## 23 5.783662e-01
## 24 5.783662e-01
## 25 1.603781e-01
## 26 1.603781e-01
## 27 1.603781e-01
## 28 1.461216e-02
## 29 1.461216e-02
## 30 1.461216e-02
## 31 1.232705e-04
## 32 1.232705e-04
## 33 1.232705e-04
## 34 8.775362e-09
## 35 8.775362e-09
## 36 8.775362e-09
```

```
iter(x5)
```

```
##    k         x             f      invDf.1     invDf.2      invDf.3
## 1  0  1.000000 -9.000000e+00  0.333333333  1.00000000 -0.333333333
## 2  0  1.000000 -3.000000e+00  0.666666667  3.00000000 -1.666666667
## 3  0 -1.000000 -1.000000e+00  0.000000000  2.00000000 -1.000000000
## 4  1  6.666667  2.500000e+01 -0.009868812  0.08216083 -0.003210336
## 5  1 14.333333  3.211111e+01 -0.004756054  0.07574016 -0.037691728
## 6  1  4.000000 -1.135556e+02  0.126828108 -0.01973762  0.005112758
## 7  2  3.910560  3.827202e+00 -0.035436060  0.15728962 -0.012449933
## 8  2  7.740029  7.596125e+00 -0.019140094  0.15401740 -0.075785086
## 9  2  2.043676 -2.827941e+01  0.258009570 -0.07616348  0.021587326
## 10 3  2.499312  4.063028e-02 -0.055506735  0.28734567 -0.038093116
## 11 3  4.500188  1.991619e+00 -0.034680532  0.30452729 -0.148794637
## 12 3  2.245246 -6.513326e+00  0.242776800 -0.13180582  0.041618547
## 13 4  1.681171  2.742807e-01 -0.076396203  0.52602059 -0.102943898
## 14 4  2.925947  6.693545e-01 -0.052942258  0.57063520 -0.277444780
## 15 4  2.768964 -1.139527e+00  0.203927971 -0.19802636  0.068687904
## 16 5  1.232724  2.399031e-02 -0.116653064  0.91138424 -0.229231613
## 17 5  2.242355  2.011052e-01 -0.082532557  0.93177630 -0.449150305
## 18 5  2.923852 -6.508760e-02  0.205069469 -0.31519387  0.116008245
## 19 6  1.037318  4.358397e-03 -0.150002701  1.23963914 -0.342661894
## 20 6  2.027716  3.818340e-02 -0.101851746  1.16900026 -0.559953238
## 21 6  2.989870  3.029681e-02  0.209349288 -0.40279998  0.150945532
## 22 7  1.001020  9.790468e-05 -0.157686329  1.31383426 -0.367790397
## 23 7  2.000488  1.317567e-03 -0.105197196  1.20972246 -0.578588722
## 24 7  2.999765  1.893787e-03  0.210497076 -0.42062574  0.157742210
## 25 8  1.000001  5.516108e-08 -0.157894607  1.31578829 -0.368420690
## 26 8  2.000000  1.038586e-06 -0.105263130  1.21052597 -0.578947221
## 27 8  3.000000  1.839156e-06  0.210526299 -0.42105240  0.157894659
##             err
## 1  1.333333e+01
## 2  1.333333e+01
## 3  1.333333e+01
## 4  6.593305e+00
## 5  6.593305e+00
## 6  6.593305e+00
## 7  3.239840e+00
## 8  3.239840e+00
## 9  3.239840e+00
## 10 1.574241e+00
## 11 1.574241e+00
## 12 1.574241e+00
## 13 6.835920e-01
## 14 6.835920e-01
## 15 6.835920e-01
## 16 2.146392e-01
## 17 2.146392e-01
## 18 2.146392e-01
## 19 3.629831e-02
## 20 3.629831e-02
## 21 3.629831e-02
## 22 1.019110e-03
## 23 1.019110e-03
## 24 1.019110e-03
```

```
## 25 6.802660e-07
## 26 6.802660e-07
## 27 6.802660e-07
```

```
iter(x6)
```

```
##    k          x             f     invDf.1     invDf.2      invDf.3
## 1  0  1.0000000 -1.100000e+01 -0.20000000  0.20000000  0.200000000
## 2  0 -1.0000000  1.000000e+00  0.13333333 -0.46666667  0.200000000
## 3  0  1.0000000  1.000000e+00  0.53333333  0.13333333 -0.200000000
## 4  1 -1.6000000  3.520444e+01 -0.01767736  0.80827637 -0.563150274
## 5  1  0.7333333  6.760000e+00  0.12121620 -3.39960941  1.718744739
## 6  1  6.9333333  1.051556e+01  0.06464864  0.18687498 -0.083336139
## 7  2 -0.5197885  7.090869e+00  0.23790483 -3.33077485  1.298846462
## 8  2  1.3737881  1.166857e+00 -0.14152196  2.55359230 -1.344863231
## 9  2  4.2704648  1.923531e+00  0.10579845  0.09099508  0.005387794
## 10 3 -0.8185750  7.512465e-01  0.12755593 -1.20375457  0.335425207
## 11 3  1.9845117  8.927336e-02 -0.07033565  1.00057374 -0.521767995
## 12 3  3.4037200 -1.944366e-01  0.13849255  0.02984688  0.027373401
## 13 4 -0.7417188  1.027875e-02  0.15131283 -1.42663471  0.427263149
## 14 4  1.8465758  5.906875e-03 -0.08334591  1.15712898 -0.606656638
## 15 4  3.3023358 -7.212577e-03  0.14111725  0.04080532  0.027161606
## 16 5 -0.7317655  2.236927e-06  0.15387463 -1.45352257  0.437691442
## 17 5  1.8362219  9.906846e-05 -0.08426755  1.17019315 -0.613885701
## 18 5  3.3008402  9.093406e-05  0.14093274  0.04291777  0.026689305
## 19 6 -0.7316617  4.891909e-11  0.15389783 -1.45375843  0.437776270
## 20 6  1.8361620  1.078544e-08 -0.08427208  1.17026181 -0.613925660
## 21 6  3.3008332  1.798076e-08  0.14093021  0.04294319  0.026682565
##             err
## 1  5.933333e+00
## 2  5.933333e+00
## 3  5.933333e+00
## 4  2.662869e+00
## 5  2.662869e+00
## 6  2.662869e+00
## 7  8.667448e-01
## 8  8.667448e-01
## 9  8.667448e-01
## 10 1.379360e-01
## 11 1.379360e-01
## 12 1.379360e-01
## 13 1.035388e-02
## 14 1.035388e-02
## 15 1.035388e-02
## 16 1.038530e-04
## 17 1.038530e-04
## 18 1.038530e-04
## 19 7.800348e-09
## 20 7.800348e-09
## 21 7.800348e-09
```

```
iter(x7)
```

```
##    k          x             f      invDf.1       invDf.2      invDf.3
## 1  0 -1.0000000 -1.100000e+01 -1.000000000  5.0000000000 -3.000000000
## 2  0  1.0000000 -1.000000e+00  2.000000000 -9.0000000000  5.000000000
## 3  0  1.0000000  1.000000e+00  0.000000000  2.0000000000 -1.000000000
## 4  1 -4.0000000  9.000000e+00  0.016456922 -0.1403678606  0.008712488
## 5  1  9.0000000  9.000000e+00 -0.007744434  0.1248789932 -0.062923524
## 6  1  4.0000000 -4.600000e+01  0.123910939  0.0019361084  0.006776379
## 7  2 -2.4840271  6.738932e-01  0.034999531 -0.2508255047  0.028292187
## 8  2  5.0513069  2.298174e+00 -0.019102158  0.2467549176 -0.125299958
## 9  2  3.1790900 -1.099583e+01  0.154777408  0.0006402126  0.015257160
## 10 3 -1.6200764  3.842807e-03  0.057552657 -0.4552660601  0.082200482
## 11 3  3.1193170  7.464109e-01 -0.035596952  0.4724765243 -0.241731360
## 12 3  3.2410804 -2.239763e+00  0.150882448 -0.0026550505  0.024610756
## 13 4 -1.0963724  3.194986e-03  0.094026499 -0.8270801754  0.206955740
## 14 4  2.2253712  2.742659e-01 -0.059748337  0.8153537577 -0.421300976
## 15 4  3.2976046 -2.506074e-01  0.146427779  0.0017780206  0.032500142
## 16 5 -0.8179683  5.168591e-05  0.135956997 -1.2690753090  0.370455600
## 17 5  1.8963572  7.750883e-02 -0.079641880  1.1014827405 -0.575082741
## 18 5  3.3047939  4.676749e-02  0.142775149  0.0253559788  0.030959138
## 19 6 -0.7369361  1.170032e-05  0.152815547 -1.4432028197  0.434200476
## 20 6  1.8378819  6.566224e-03 -0.084174313  1.1686743335 -0.612892667
## 21 6  3.3013733  9.713082e-03  0.141056262  0.0415779220  0.027063312
## 22 7 -0.7316789  2.889369e-07  0.153894642 -1.4537293089  0.437767423
## 23 7  1.8361622  2.763795e-05 -0.084272301  1.1702644251 -0.613926368
## 24 7  3.3008358  5.231847e-05  0.140930618  0.0429383496  0.026683991
## 25 8 -0.7316617  6.881606e-12  0.153897836 -1.4537584449  0.437776274
## 26 8  1.8361620  2.968834e-10 -0.084272079  1.1702618080 -0.613925661
## 27 8  3.3008332  5.937273e-10  0.140930212  0.0429431936  0.026682564
##             err
## 1  8.000000e+00
## 2  8.000000e+00
## 3  8.000000e+00
## 4  3.948693e+00
## 5  3.948693e+00
## 6  3.948693e+00
## 7  1.931990e+00
## 8  1.931990e+00
## 9  1.931990e+00
## 10 8.939458e-01
## 11 8.939458e-01
## 12 8.939458e-01
## 13 3.290139e-01
## 14 3.290139e-01
## 15 3.290139e-01
## 16 8.103224e-02
## 17 8.103224e-02
## 18 8.103224e-02
## 19 5.257181e-03
## 20 5.257181e-03
## 21 5.257181e-03
## 22 1.723031e-05
## 23 1.723031e-05
## 24 1.723031e-05
```

```
## 25 1.706180e-10
## 26 1.706180e-10
## 27 1.706180e-10
```

```
iter(x8)
```

```
##    k        x             f     invDf.1    invDf.2     invDf.3
## 1  0 1.000000 -9.000000e+00  0.33333333 -1.6666667  1.00000000
## 2  0 1.000000 -1.000000e+00  0.66666667 -2.3333333  1.00000000
## 3  0 1.000000  1.000000e+00  0.00000000  2.0000000 -1.00000000
## 4  1 1.333333  9.000000e+00 -0.06044539  0.5694592 -0.08589608
## 5  1 3.666667  1.111111e-01 -0.02545069  0.3976670 -0.19406151
## 6  1 4.000000 -6.888889e+00  0.13573701 -0.1208908  0.03499470
## 7  2 1.222340  9.353331e-01 -0.10480112  0.8194737 -0.18377661
## 8  2 2.514669  1.231951e-02 -0.06358492  0.7453707 -0.35968059
## 9  2 3.032874 -1.302458e+00  0.19262029 -0.2579805  0.08959443
## 10 3 1.070907  3.635235e-03 -0.14134875  1.1477936 -0.30745239
## 11 3 2.096491  2.293194e-02 -0.09481505  1.0831131 -0.51942227
## 12 3 2.972581 -1.290094e-01  0.20792770 -0.3752474  0.13908363
## 13 4 1.005436  6.652470e-04 -0.15663242  1.3033802 -0.36409521
## 14 4 2.004987  4.286524e-03 -0.10464128  1.2029767 -0.57546945
## 15 4 2.998373  2.001443e-04  0.21032633 -0.4179528  0.15667907
## 16 5 1.000026  2.625331e-06 -0.15788927  1.3157374 -0.36840379
## 17 5 2.000016  2.926701e-05 -0.10526112  1.2105015 -0.57893615
## 18 5 2.999994  3.381524e-05  0.21052552 -0.4210407  0.15789033
## 19 6 1.000000  4.135359e-11 -0.15789474  1.3157895 -0.36842105
## 20 6 2.000000  6.571810e-10 -0.10526316  1.2105263 -0.57894737
## 21 6 3.000000  1.071795e-09  0.21052632 -0.4210526  0.15789474
##             err
## 1  3.000000e+00
## 2  3.000000e+00
## 3  3.000000e+00
## 4  1.151997e+00
## 5  1.151997e+00
## 6  1.151997e+00
## 7  4.181785e-01
## 8  4.181785e-01
## 9  4.181785e-01
## 10 9.150357e-02
## 11 9.150357e-02
## 12 9.150357e-02
## 13 5.409900e-03
## 14 5.409900e-03
## 15 5.409900e-03
## 16 2.563553e-05
## 17 2.563553e-05
## 18 2.563553e-05
## 19 4.633104e-10
## 20 4.633104e-10
## 21 4.633104e-10
```

Attempt 3 gives us a $(-,+,+)$ solution while we are asking for $(-,+,-)$. Attempt 4 gives us a $(-,+,+)$ solution while we are asking for $(-,-,+)$.

What's more, attempt 5 and 8 are giving the same result, while attempt 6 and 7 are the same. This should be fixed. The solution we got in attempt 5 is $(+,+,+)$ while we are trying to find a $(+,+,-)$ solution. The solution in attempt 6 is $(-,+,+)$ while we are trying to get $(+,-,+)$. So in this case, we should rerun attempt 3, 4, 5, 6 with x_1, x_2, x_3 stay the same sign but different values.

```
x3 <- c(-3, 3, -3)
x5 <- c(2, 2, -2)
iter(x3)
```

```
##    k         x             f     invDf.1     invDf.2     invDf.3
## 1  0 -3.000000 -3.000000e+00 -0.03448276 -0.24137931  0.03448276
## 2  0  3.000000  1.000000e+00  0.04137931  0.48965517 -0.24137931
## 3  0 -3.000000  5.000000e+00 -0.16551724  0.04137931 -0.03448276
## 4  1 -3.034483  1.336029e-01 -0.02886356 -0.21903457  0.02475296
## 5  1  3.841379  1.189061e-03  0.02621252  0.34855565 -0.17211827
## 6  1 -3.365517 -7.055410e-01 -0.14895943  0.01924237 -0.02189341
## 7  2 -3.012902  1.964106e-05 -0.02928624 -0.22292117  0.02605403
## 8  2  3.716026  4.657364e-04  0.02743652  0.36431282 -0.17987982
## 9  2 -3.361085 -1.478191e-02 -0.14903664  0.02103363 -0.02288335
## 10 3 -3.012412  1.191137e-07 -0.02929218 -0.22301265  0.02608540
## 11 3  3.713197  2.396357e-07  0.02746178  0.36468540 -0.18006365
## 12 3 -3.361431 -7.524985e-06 -0.14901846  0.02107328 -0.02290368
##             err
## 1  8.413793e-01
## 2  8.413793e-01
## 3  8.413793e-01
## 4  1.253530e-01
## 5  1.253530e-01
## 6  1.253530e-01
## 7  2.829179e-03
## 8  2.829179e-03
## 9  2.829179e-03
## 10 1.445639e-06
## 11 1.445639e-06
## 12 1.445639e-06
```

```
iter(x5)
```

```
##    k         x             f     invDf.1   invDf.2     invDf.3
## 1  0  2.000000 -4.000000e+00  0.07692308 0.3846154 -0.07692308
## 2  0  2.000000 -2.000000e+00  0.10256410 0.8461538 -0.43589744
## 3  0 -2.000000  1.000000e+00 -0.20512821 0.3076923 -0.12820513
## 4  1  3.153846  5.917160e-03  0.04042335 0.1836967 -0.01578435
## 5  1  4.538462  1.331361e+00  0.03156871 0.2672679 -0.13613635
## 6  1 -2.076923 -3.781065e+00 -0.22340932 0.1085656 -0.03657350
## 7  2  2.849359  7.924495e-02  0.04075527 0.2161470 -0.02391029
## 8  2  3.667704  9.271263e-02  0.03665947 0.3522678 -0.17935039
## 9  2 -2.358428 -5.727929e-01 -0.19559327 0.1205072 -0.04309241
## 10 3  2.812394  4.143580e-04  0.04119988 0.2215626 -0.02555141
## 11 3  3.529409  1.366400e-03  0.03824848 0.3707394 -0.18876970
## 12 3 -2.378784 -1.639282e-02 -0.19349209 0.1244968 -0.04504847
## 13 4  2.811655  6.862610e-07  0.04120413 0.2216956 -0.02559587
## 14 4  3.525792  5.456396e-07  0.03828809 0.3712520 -0.18903043
## 15 4 -2.379612 -1.199064e-05 -0.19341551 0.1245891 -0.04509691
##             err
## 1  2.538462e+00
## 2  2.538462e+00
## 3  2.538462e+00
## 4  8.707573e-01
## 5  8.707573e-01
## 6  8.707573e-01
## 7  1.382954e-01
## 8  1.382954e-01
## 9  1.382954e-01
## 10 3.616894e-03
## 11 3.616894e-03
## 12 3.616894e-03
## 13 2.495442e-06
## 14 2.495442e-06
## 15 2.495442e-06
```

Now we got $(-,+,-)$ from attempt 3 and $(+,+,-)$ from attempt 5.

While for Attempt 4 and Attempt 6, no matter how many times I tried with different values, I still cannot get $(-.-,+)$ and $(+,-,+)$. So it could be considered that try to get these two solutions may cause the iterration to diverge. This is because these two solutions are complex solutions.

Finally, we got 6 solutions and they are:

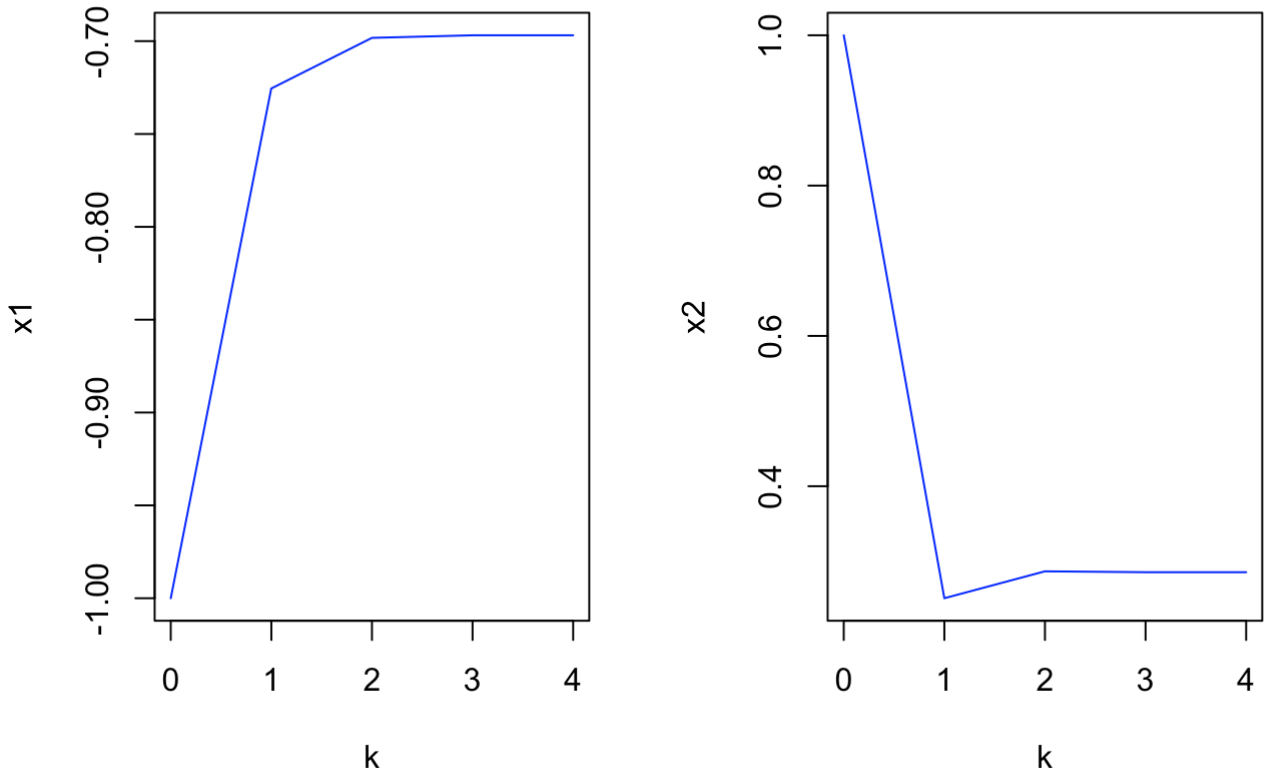
$$(-2.0390182646380097, -1.8254237212876712, -3.983019204815071)$$
$$(1.9257450572512143, -1.7267676182142317, -3.4352616437417134)$$
$$(-3.0124120515940755, 3.713195662979653, -3.36143070560951)$$
$$(2.8116546777359908, 3.5257894881899334, -2.3796125386445377)$$
$$(-0.7316616530662688, 1.8361619567132068, 3.3008331821455417)$$
$$(1.000000000000278, 2.0000000000000426, 2.9999999999999494)$$

3. In the example above, a solution for a system of two non-linear equations was found from a starting value $x0 = (1, 1)$. Find a different solution assuming $x0 = c(-1, 1)$. Prepare plots showing the conversion of each variable as well as a table with all key components of Newton-Raphson method for each iteration.

```
tol <- 10 ^ -10
err <- 10 ^ 5
x0 <- c(-1, 1)
x <- NULL
f <- function(x){
  matrix(c(log(x[1] ^ 2 + 2 * x[2] ^ 2 + 1) - 0.5, x[2] - x[1] ^ 2 + 0.2))
}
Df <- function(x){
  matrix(c(2 * x[1] / (x[1] ^ 2 + 2 * x[2] ^ 2 + 1), 4 * x[2] / (x[1] ^ 2 + 2 * x[2] ^ 2
 + 1),
           -2 * x[1], 1),
         byrow = TRUE, nrow = length(x))
}
k <- 0
dta <- NULL
while (err>tol){
    x <- x0 - solve(Df(x0)) %*% f(x0)
    err <- max(abs(x - x0))
    dta <- rbind(dta, data.frame(k = rep(k, length(x0)), x = x0, f = f(x0), invDf = solv
e(Df(x0)), err = err))
    x0 <- x
    k <- k + 1
}
dta
```

```
##    k          x             f    invDf.1   invDf.2          err
## 1  0 -1.0000000  8.862944e-01 -0.4000000 0.4000000 7.490355e-01
## 2  0  1.0000000  2.000000e-01  0.8000000 0.2000000 7.490355e-01
## 3  1 -0.7254823  2.162734e-03 -0.5682805 0.3452618 3.582416e-02
## 4  1  0.2509645 -7.535999e-02  0.8245549 0.4990373 3.582416e-02
## 5  2 -0.6982343  2.002778e-03 -0.5509631 0.3825846 1.387507e-03
## 6  2  0.2867887 -7.424518e-04  0.7694027 0.4657326 1.387507e-03
## 7  3 -0.6968468  8.921806e-07 -0.5521855 0.3826007 1.229223e-06
## 8  3  0.2855935 -1.925176e-06  0.7695774 0.4667719 1.229223e-06
## 9  4 -0.6968456  5.729861e-13 -0.5521858 0.3826015 8.945067e-13
## 10 4  0.2855937 -1.511014e-12  0.7695765 0.4667717 8.945067e-13
```

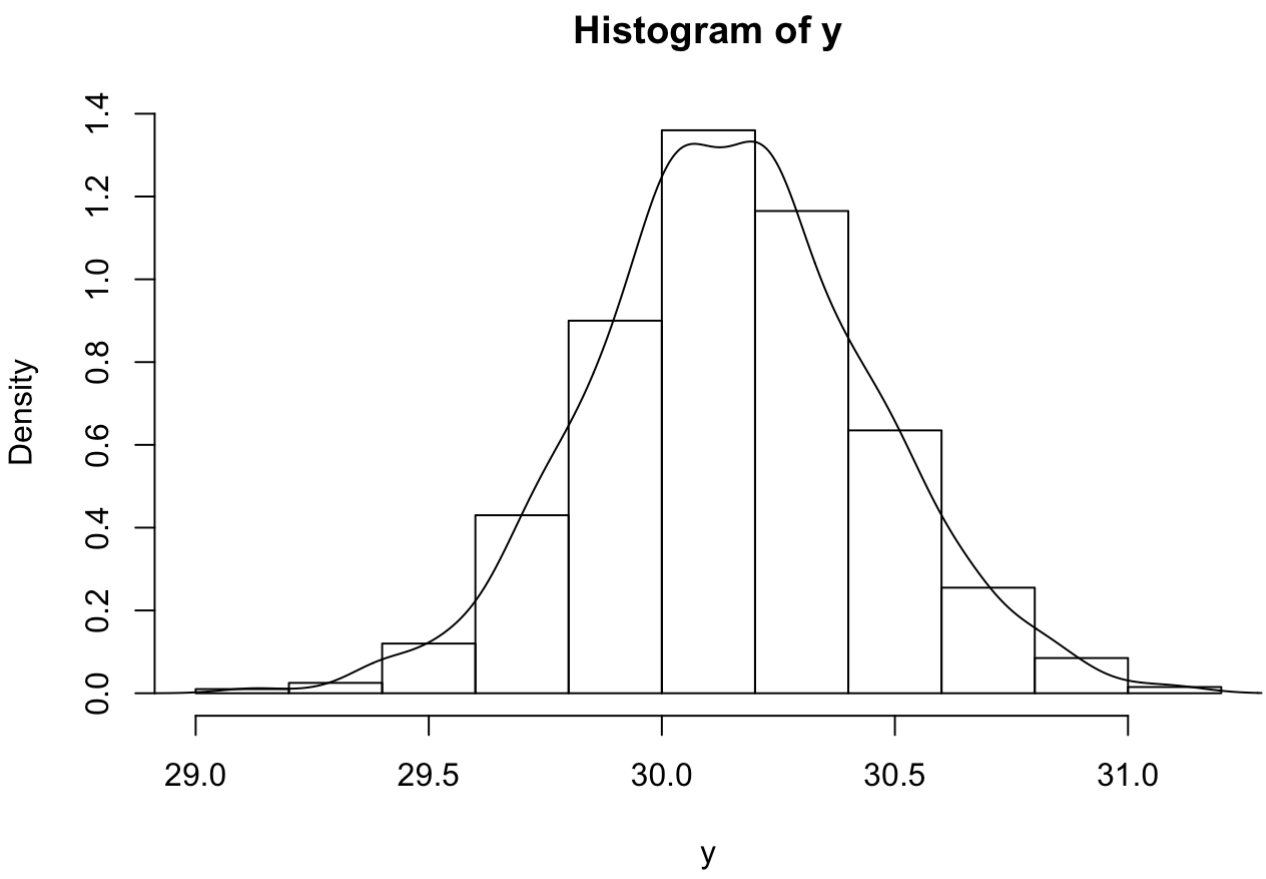
```
dta1 <- aggregate(x ~ k, dta, FUN = head, 1)
dta2 <- aggregate(x ~ k, dta, FUN = tail, 1)
par(mfrow = c(1, 2))
plot(dta1$k, dta1$x, type = "l", col = "blue", xlab = "k", ylab = "x1")
plot(dta2$k, dta2$x, type = "l", col = "blue", xlab = "k", ylab = "x2")
```



4. The data file “z1036.rnorm.RData” contains a random sample, y, from a normal distribution.

- Write down the log-likelihood function, l, given the sample.

```
dataPath <- "/Users/doubao/Desktop/Course Work/Linear & Non-Linear"
load(paste(dataPath, "z1036.rnorm.rdata", sep = "/"))
y <- dta
hist(y, probability = T)
lines(density(y))
```




```
c(mean = mean(y), sd = sd(y))
```

```
##       mean         sd
## 30.1514433  0.3021608
```

The likelihood function can be written as:

$$\begin{aligned} L(y;\lambda) &= \prod_{i=1}^{n} \frac{1}{(2\pi)^{\frac{1}{2}}\sigma} e^{-\frac{1}{2\sigma^2}(y_i-\mu)^2} \\ &= \frac{1}{(2\pi)^{\frac{n}{2}}\sigma^n} e^{-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i-\mu)^2} \end{aligned}$$

The log-likelihood function can be written as:

$$l(y;\lambda) = -\frac{n}{2}\log 2\pi - n\log\sigma - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i-\mu)^2$$

- Derive analytically:

$$U = \begin{bmatrix} \frac{\partial l}{\partial \mu} \\ \frac{\partial l}{\partial \sigma} \end{bmatrix}$$

```
U <- function(x){ # mu as x[1], sigma as x[2]
  matrix(c(sum(y) / x[2] ^ 2 - n * x[1] / x[2] ^ 2,
           - n / x[2] + 1 / x[2] ^3 * sum((y - x[1]) ^ 2)))
}
```

- Derive analytically:

$$U' = \begin{bmatrix} \frac{\partial^2 l}{\partial \mu^2} & \frac{\partial^2 l}{\partial \mu \partial \sigma} \\ \frac{\partial^2 l}{\partial \sigma \partial \mu} & \frac{\partial^2 l}{\partial \sigma^2} \end{bmatrix}$$

```
dU <- function(x){
  matrix(c(- n / x[2] ^ 2, - 2 * (sum(y) - n * x[1]) / x[2] ^ 3,
           (2 * n * x[1] - 2 * sum(y)) / x[2] ^ 3, n / x[2] ^ 2 - 3 * sum((y - x[1]) ^
2) /x[2] ^ 4),
         byrow = T, nrow = 2)
}
```

- Implement the Newton-Raphson algorithm in R and find the solution for (σ, μ). Plot the value of the xmeters at each iteration.

```
err <- 10 ^ 10
tol <- 10 ^ - 10
x0 <- c(30, 0.1)
x <- NULL
k <- 0
dta <- NULL
(n <- length(y))
```

```
## [1] 1000
```

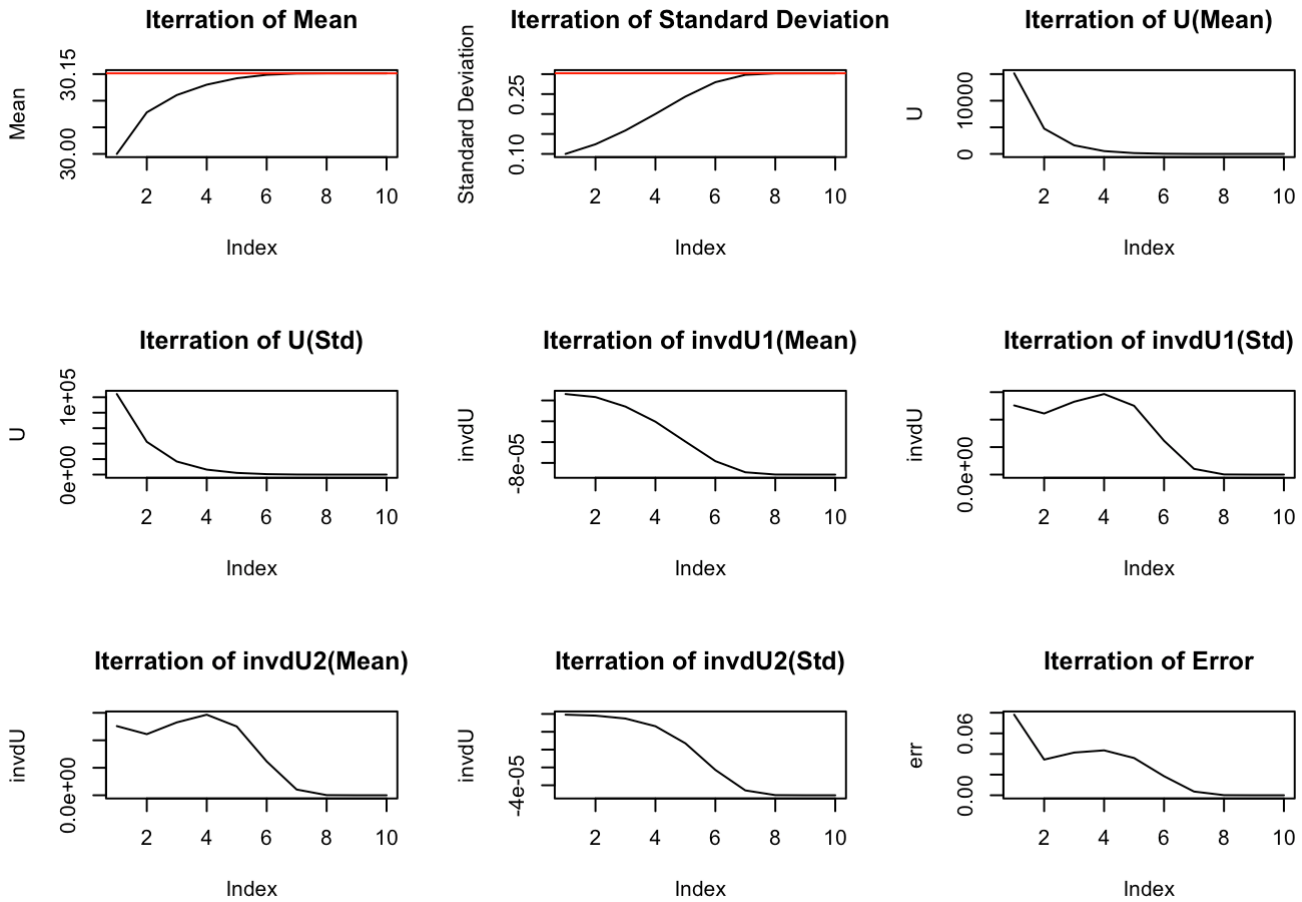
```
while (err > tol){
  x <- x0 - solve(dU(x0)) %*% U(x0)
  err <- max(abs(x - x0))
  dta <- rbind(dta, data.frame(k = rep(k, 2), x = x0, U = U(x0), invdU = solve(dU(x0)),e
rr = err))
  x0 <- x
  k <- k + 1
}
dta
```

```
##    k          x            U       invdU.1       invdU.2          err
## 1  0 30.0000000 1.514433e+04 -1.381148e-05  1.258386e-06 7.811113e-02
## 2  0  0.1000000 1.041449e+05  1.258386e-06 -4.154644e-07 7.811113e-02
## 3  1 30.0781111 4.753065e+03 -1.674106e-05  1.111702e-06 3.458910e-02
## 4  1  0.1242111 4.235024e+04  1.111702e-06 -9.415080e-07 3.458910e-02
## 5  2 30.1106016 1.619576e+03 -2.589941e-05  1.325684e-06 4.139802e-02
## 6  2  0.1588002 1.689591e+04  1.325684e-06 -2.577256e-06 4.139802e-02
## 7  3 30.1301491 5.313021e+02 -4.039119e-05  1.466008e-06 4.352468e-02
## 8  3  0.2001982 6.428864e+03  1.466008e-06 -6.891353e-06 4.352468e-02
## 9  4 30.1421842 1.558744e+02 -5.949597e-05  1.251937e-06 3.610527e-02
## 10 4  0.2437229 2.203080e+03  1.251937e-06 -1.647712e-05 3.610527e-02
## 11 5 30.1487000 3.503392e+01 -7.831587e-05  6.155070e-07 1.847945e-02
## 12 5  0.2798282 5.893493e+02  6.155070e-07 -3.139228e-05 1.847945e-02
## 13 6 30.1510810 4.071651e+00 -8.898769e-05  1.041818e-07 3.590350e-03
## 14 6  0.2983076 8.372618e+01  1.041818e-07 -4.288711e-05 3.590350e-03
## 15 7 30.1514346 9.569324e-02 -9.114238e-05  2.630145e-09 1.116436e-04
## 16 7  0.3018980 2.452599e+00  2.630145e-09 -4.552063e-05 1.116436e-04
## 17 8 30.1514433 7.072347e-05 -9.120981e-05  1.948165e-12 1.034414e-07
## 18 8  0.3020096 2.268210e-03  1.948165e-12 -4.560486e-05 1.034414e-07
## 19 9 30.1514433 5.820766e-11 -9.120987e-05  2.197410e-18 8.865131e-14
## 20 9  0.3020097 1.943590e-09  2.197410e-18 -4.560493e-05 8.865131e-14
```

```
res <- (rbind(c(mean = x0[1], sd = x0[2]),
      c(mean(y),  sd(y))))
rownames(res) <- c("Newton-Raphson", "Sample Statistics")
res
```

```
##                       mean        sd
## Newton-Raphson    30.15144 0.3020097
## Sample Statistics 30.15144 0.3021608
```

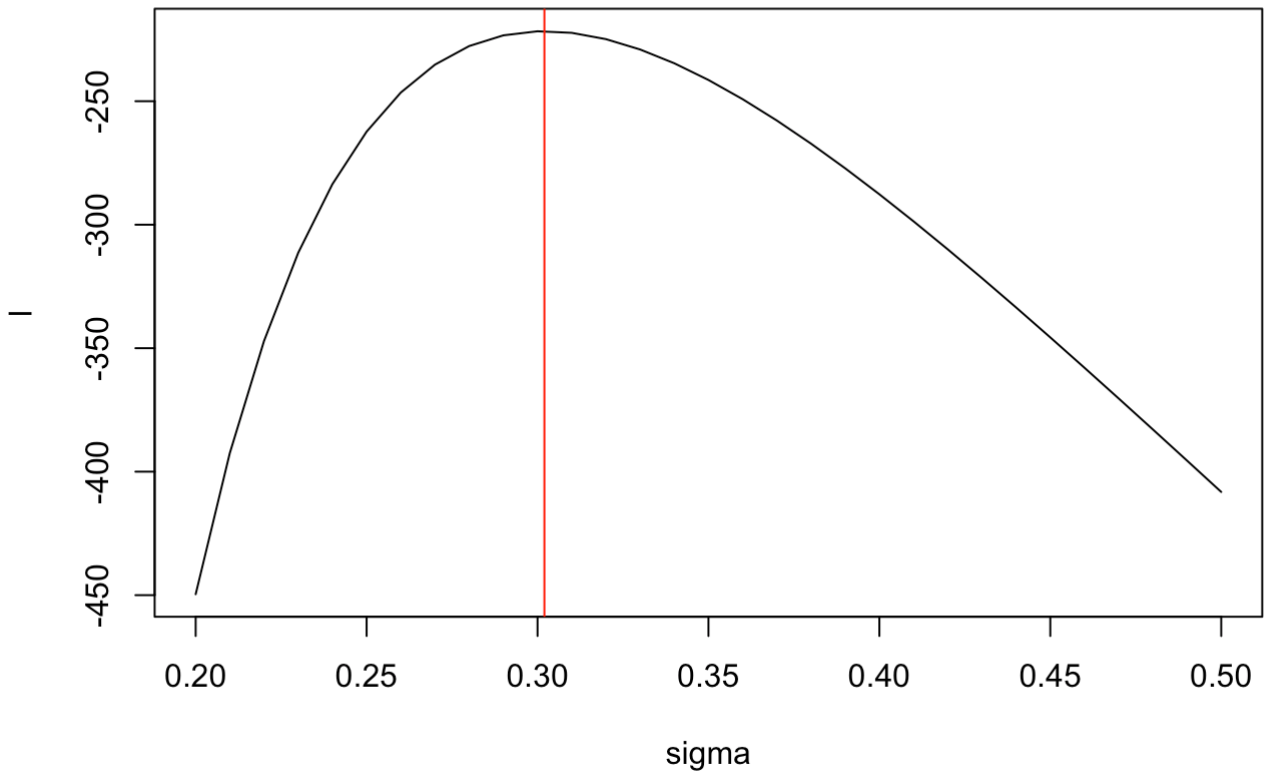
```
par(mfrow = c(3, 3))
plot(dta$x[seq(1, length(dta$x), by = 2)], type = "l", main = "Iterration of Mean", ylab
 = "Mean")
abline(h = mean(y), col = "red")
plot(dta$x[seq(2, length(dta$x), by = 2)], type = "l", main = "Iterration of Standard De
viation", ylab = "Standard Deviation")
abline(h = sd(y), col = "red")
plot(dta$U[seq(1, length(dta$x), by = 2)], type = "l", main = "Iterration of U(Mean)", y
lab = "U")
plot(dta$U[seq(2, length(dta$x), by = 2)], type = "l", main = "Iterration of U(Std)", yl
ab = "U")
plot(dta$invdU.1[seq(1, length(dta$x), by = 2)], type = "l", main = "Iterration of invdU
1(Mean)", ylab = "invdU")
plot(dta$invdU.1[seq(2, length(dta$x), by = 2)], type = "l", main = "Iterration of invdU
1(Std)", ylab = "invdU")
plot(dta$invdU.2[seq(1, length(dta$x), by = 2)], type = "l", main = "Iterration of invdU
2(Mean)", ylab = "invdU")
plot(dta$invdU.2[seq(2, length(dta$x), by = 2)], type = "l", main = "Iterration of invdU
2(Std)", ylab = "invdU")
plot(dta$err[seq(1, length(dta$x), by = 2)], type = "l", main = "Iterration of Error", y
lab = "err")
```



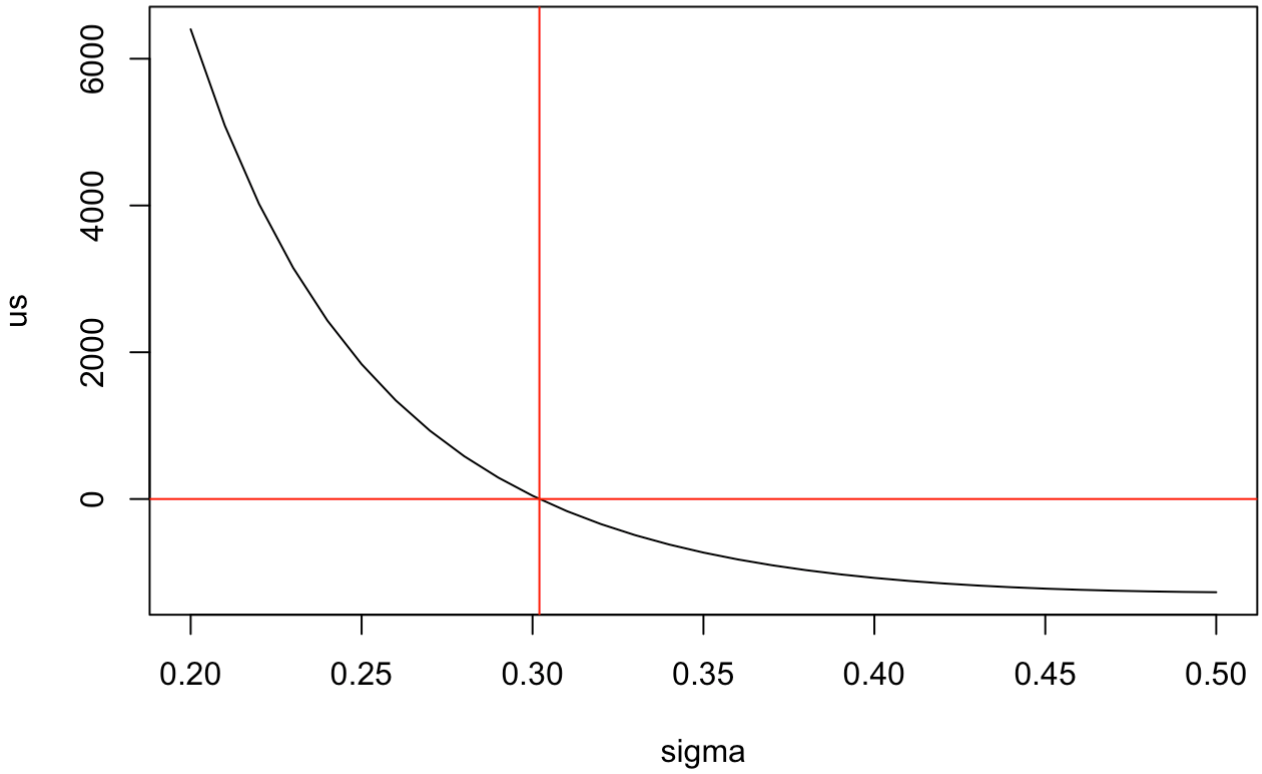
```
# U <- function(e, s){
#   matrix(c( (sum(y) - n*e)/(s^2),
#             -n/s + sum(y^2)/(s^3) + n*e^2/(s^3) - 2*e*sum(y)/(s^3)))}
# dU <- function(e, s){
#   matrix(c(-n/(s^2),
#            2*(n*e-sum(y))/(s^3),
#            2*(n*e-sum(y))/(s^3),
#            n/(s^2) - 3*sum(y^2)/(s^4) - 3*n*e^2/(s^4) + 6*e*sum(y)/(s^4)),
#          byrow = T, nrow = 2)
# }
# err <- 10 ^ 10
# tol <- 10 ^ - 10
# e0 <- 10
# s0 <- 0.2
# e <- NULL
# s <- NULL
# k <- 0
# dta <- NULL
# (n <- length(y))
# x <- NULL
# while (err > tol){
#   e <- (matrix(c(e0, s0)) - solve(dU(e0, s0)) %*% U(e0, s0))[1]
#   s <- (matrix(c(e0, s0)) - solve(dU(e0, s0)) %*% U(e0, s0))[2]
#   err <- max(abs(e - e0), abs(s - s0))
#   dta <- rbind(dta, data.frame(k = k, e = e0, s = s0, U = U(e0, s0), invdU = solve(dU
(e0, s0)),err = err))
#   e0 <- e
#   s0 <- s
#   k <- k + 1
# }
# dta
```

- Plot the log-likelihood function and its first derivative w.r.t. σ for a range of values for σσ in the vicinity of the solution, . Fix μ at the solution.

```
mu <- dta$x[length(dta$x) - 1]
sigma <- seq(0.2, 0.5, by = 0.01)
l <- (-n / 2) * log(2 * pi) - n * log(sigma) - sum((y - mu)^2) / (2 * sigma ^2)
us <- - n / sigma + sum(y ^ 2) / (sigma ^ 3) + n * mu ^ 2 / (sigma ^ 3) - 2 * mu *
sum(y) / (sigma ^ 3)
plot(sigma, l, type = "l")
abline(v = dta$x[length(dta$x)], col = "red")
```

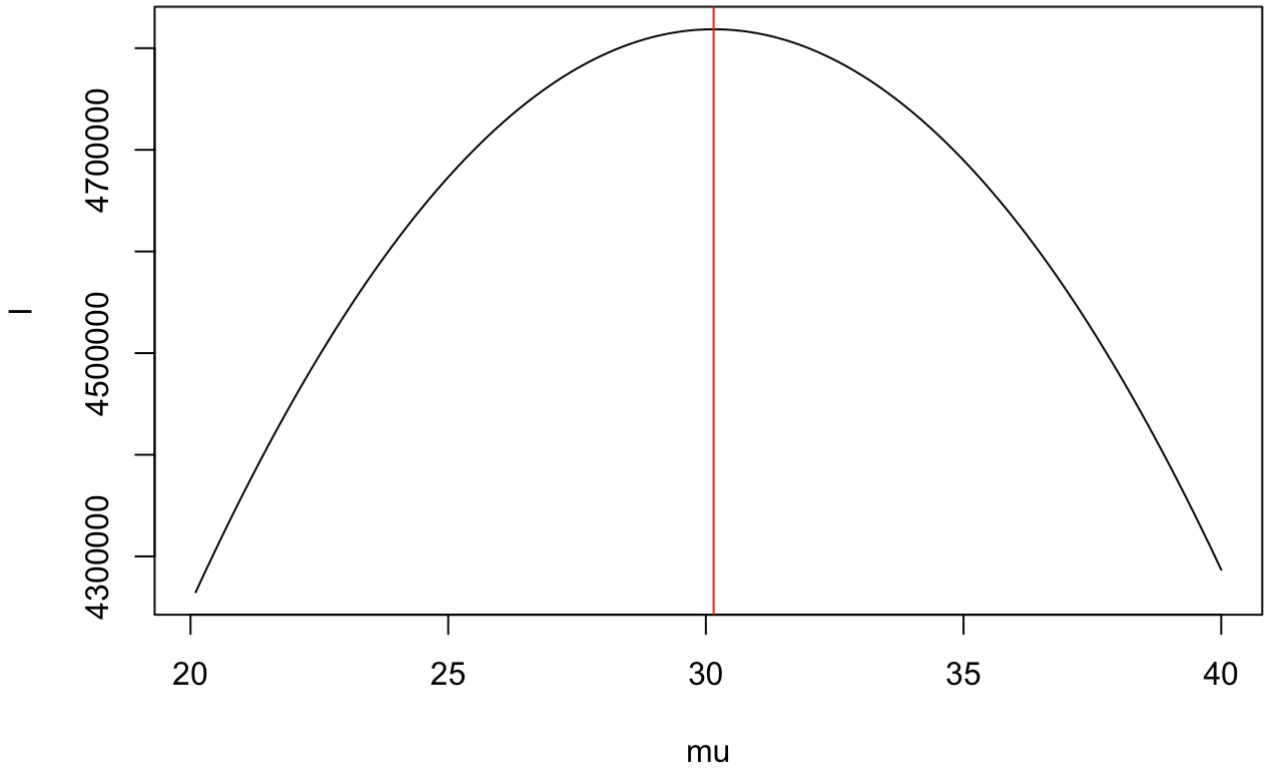



```
plot(sigma, us, type = "l")
abline(h = 0, col = "red")
abline(v = dta$x[length(dta$x)], col = "red")
```

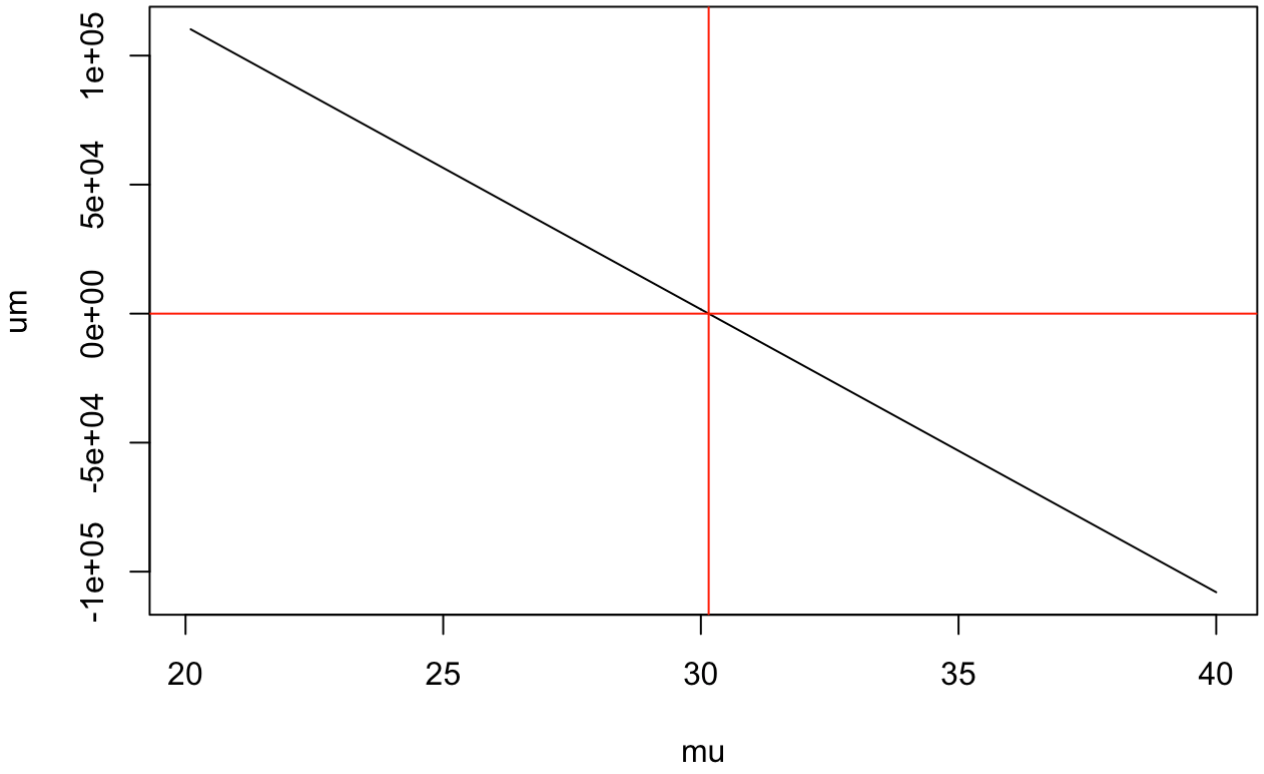


- Plot the log-likelihood function and its first derivative w.r.t. μ for a range of values for μ in the vicinity of the solution, . Fix σ at the solution.

```
sigma <- dta$x[length(dta$x)]
mu <- seq(20.1, 40, by = 0.1)
l <- (-n / 2) * log(2 * pi) - n * log(sigma) - (sum(y) + n * mu ^ 2 - 2 * sum(y) * mu) /
 (2 * sigma ^2)
um <- (sum(y) - n * mu) / sigma ^ 2
plot(mu, l, type = "l")
abline(v = dta$x[length(dta$x) - 1], col = "red")
```




```
plot(mu, um, type = "l")
abline(h = 0, col = "red")
abline(v = dta$x[length(dta$x) - 1], col = "red")
```


um
1e+05
5e+04
0e+00
-5e+04
-1e+05
20
25
30
35
40
mu