Problem Set 6

YSC4204 – Statistical Computing

Due on: Friday, 18 November, 23:59

Problem 1

Golden section search

Write your own R program for golden section search of a local minimum.[1] Arguments should be supplied in the form

```
gss_min <- function(f, lower, upper, tol=1e-4)
```

`f` is the function whose minimum we seek. `lower` and `upper` are the lower and upper end points of the search interval. The return value of `gss_min()` should be guaranteed to be in the interval [`lower`, `upper`], including the boundaries.

(6 points.)

Problem 2

Nelder-Mead algorithm

Consider the bivariate objective function from Givens & Hoeting, page 50,

$$g(x_1, x_2) = \begin{cases} -360x_1^2 - x_2 - x_2^2 & \text{if } x_1 \leq 0, \\ -6x_1^2 - x_2 - x_2^2 & \text{otherwise.} \end{cases}$$

Givens & Hoeting use g in an example where the Nelder-Mead method fails. However, it is impossible (or at least difficult) to start R's `optim()` with the particular initial simplex chosen by Givens & Hoeting. Let us instead run the code in Fig. 1, which evaluates g starting from a different simplex.

Explain the lines of output shown in Fig 1. Why does R evaluate g at these particular coordinates? Why does the Nelder-Mead algorithm decide to perform "EXTENSION", "REFLECTION" and "HI-REDUCTION" based on the calculated values of g?

Incidentally, with the chosen initial condition, Nelder-Mead succeeds in finding the maximum. The first steps in the algorithm are shown in Fig. 2.

(5 points.)

[1] It is acceptable to translate code from Numerical Recipes into R.

```
g <- function(x) {
  if(x[1] <= 0) {
    val <- -360 * x[1]^2 - x[2] - x[2]^2
  } else {
    val <- -6 * x[1]^2 - x[2] - x[2]^2
  }
  cat("evaluating g at x1 = ", x[1], ", x2 = ", x[2],
      ", g = ", val, "\n", sep = "")
  return(val)
}
o <- optim(c(1, 0), function(x) -g(x),
           control = list(trace = 1, maxit = 13))
```

Output:

```
  Nelder-Mead direct search function minimizer
evaluating g at x1 = 1, x2 = 0, g = -6
function value for initial parameters = 6.000000
  Scaled convergence tolerance is 8.9407e-08
Stepsize computed as 0.100000
evaluating g at x1 = 1.1, x2 = 0, g = -7.26
evaluating g at x1 = 1, x2 = 0.1, g = -6.11
BUILD              3 7.260000 6.000000
evaluating g at x1 = 0.9, x2 = 0.1, g = -4.97
evaluating g at x1 = 0.8, x2 = 0.15, g = -4.0125
EXTENSION          5 6.110000 4.012500
evaluating g at x1 = 0.8, x2 = 0.05, g = -3.8925
evaluating g at x1 = 0.7, x2 = 0.025, g = -2.965625
EXTENSION          7 6.000000 2.965625
evaluating g at x1 = 0.5, x2 = 0.175, g = -1.705625
evaluating g at x1 = 0.25, x2 = 0.2625, g = -0.7064062
EXTENSION          9 4.012500 0.706406
evaluating g at x1 = 0.15, x2 = 0.1375, g = -0.2914062
evaluating g at x1 = -0.175, x2 = 0.13125, g = -11.17348
REFLECTION        11 2.965625 0.291406
evaluating g at x1 = -0.3, x2 = 0.375, g = -32.91563
evaluating g at x1 = 0.45, x2 = 0.1125, g = -1.340156
HI-REDUCTION 13 1.340156 0.291406
```

Figure 1: Running Nelder-Mead algorithm in R.

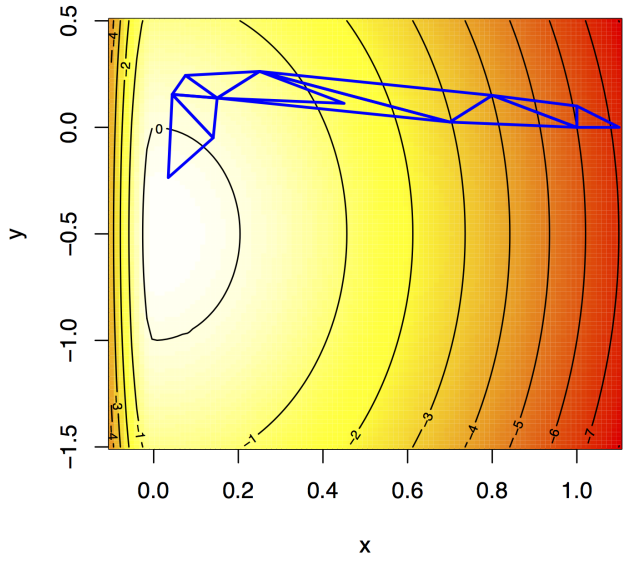

Figure 2: The first 10 simplices of the Nelder-Mead algorithm when applied to the function g in problem 2.

Problem 3

Simpson and Gauss-Legendre integration

(a) Approximate the integral $\int_{-1}^{1} x^4\,dx$ with the Simpson rule, evaluating the integrand only at $x = 0, \pm 0.2, \ldots, \pm 1$. Show your R code. How much does the Simpson quadrature differ from the exact integral $\frac{2}{5}$?

(b) Now approximate the same integral with the 10-point Gauss-Legendre quadrature. The nodes and weights are given in Table 1. (You can find similar tables online if you want to copy and paste the numbers into your code.) Again show your R code and comment on the accuracy of the result in comparison to (a).

(4 points. 2 each for (a) and (b).)

i	weight $-w_i$	abscissa $-x_i$
1	0.2955242247147529	-0.1488743389816312
2	0.2955242247147529	0.1488743389816312
3	0.2692667193099963	-0.4333953941292472
4	0.2692667193099963	0.4333953941292472
5	0.2190863625159820	-0.6794095682990244
6	0.2190863625159820	0.6794095682990244
7	0.1494513491505806	-0.8650633666889845
8	0.1494513491505806	0.8650633666889845
9	0.0666713443086881	-0.9739065285171717
10	0.0666713443086881	0.9739065285171717

Table 1: Weights and nodes (also called abscissae) for 10-point Gauss-Legendre quadrature in the range $[-1, 1]$