Homework 2

Elliott Pryor

23 September 2021

Table of Contents


```
clc;
```

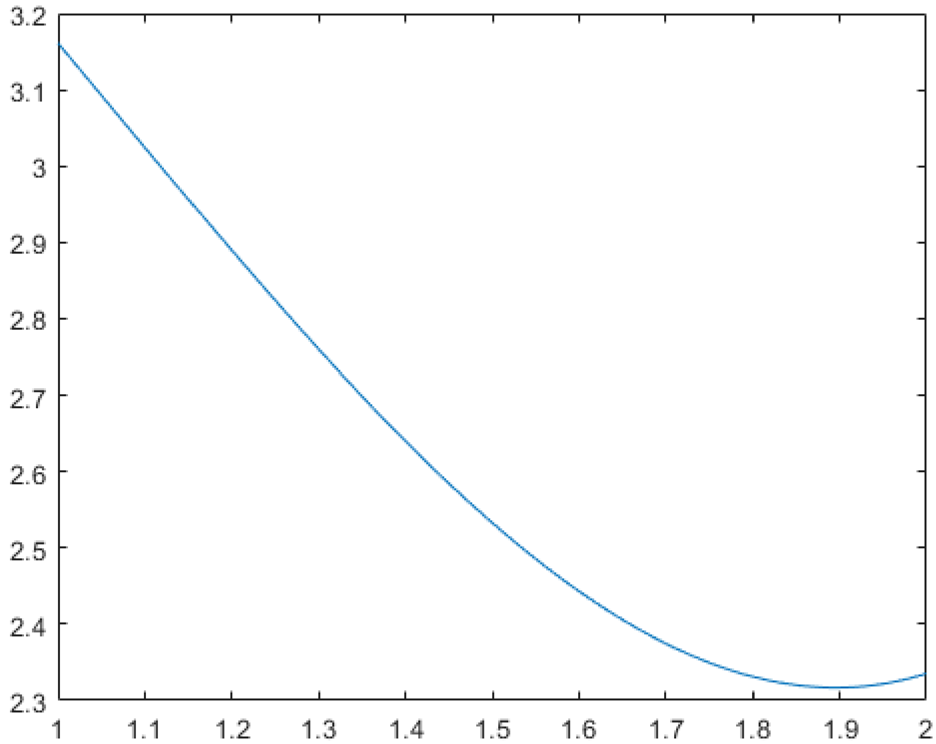
Define the function we are optimizing 1.1

```
f = @(x) x.^2 + 4 * cos(x); % lambda function syntax
f1= @(x) 2 * x - 4 * sin(x);
f2= @(x) 2 - 4 * cos(x);
a = 1;
b = 2;

eps = 0.02;  % epsilon
```

Plot the function

```
x = a:0.01:b;
plot(x, f(x));
```



Run Golden Section 1.1

```
o = golden_sec(f, 1, 2, 0.02);
fprintf("Golden Section\n")
for row=1:size(o,1)
    fprintf("k = %i,\t ak = %0.12f,\t bk = %0.12f,\t f(ak) = %0.12f,\t
 f(bk) = %0.12f\t [%0.6f, %0.6f]\n",o(row, :))
end
% This outputs at the very bottom for some strange reason
```

Run Newton 1.1

```
o = newtons(f, f1, f2, 1, 10);
fprintf("Newtons:\n")
for row=1:size(o,1)
    fprintf("k = %i,\t xk = %0.12f,\t f(xk) = %0.12f,\t f'(xk) =
 %0.12f,\t f''(xk) = %0.12f\n",o(row, :))
end
```

```
Newtons:
k = 0,  xk = 1.000000000000,  f(xk) = 3.161209223473,  f'(xk) =
 -1.365883939232,  f''(xk) = -0.161209223473
k = 1,  xk = -7.472740639831,  f(xk) = 57.330143273665,  f'(xk) =
 -11.232666837815,  f''(xk) = 0.511709396520
```

```
k = 2,  xk = 14.478520982874,  f(xk) = 208.288516590770,  f'(xk) =
 25.187832981772,  f''(xk) = 3.339053260748
k = 3,  xk = 6.935115408046,  f(xk) = 51.275482705508,  f'(xk) =
 11.443343617834,  f''(xk) = -1.179656982590
k = 4,  xk = 16.635684121431,  f(xk) = 274.347348438923,  f'(xk) =
 36.472389478119,  f''(xk) = 4.398637749105
k = 5,  xk = 8.343937549316,  f(xk) = 67.738945865403,  f'(xk) =
 13.158460678096,  f''(xk) = 3.882347961485
k = 6,  xk = 4.954632724341,  f(xk) = 25.507911272968,  f'(xk) =
 13.792474194376,  f''(xk) = 1.040474160139
k = 7,  xk = -8.301317997722,  f(xk) = 67.181618377811,  f'(xk) =
 -12.996226008912,  f''(xk) = 3.730262121486
k = 8,  xk = -4.817319933871,  f(xk) = 23.625525354616,  f'(xk) =
 -13.612639055533,  f''(xk) = 1.581045990659
k = 9,  xk = 3.792574487895,  f(xk) = 11.201664357595,  f'(xk) =
 10.009019920966,  f''(xk) = 5.181956888635
k = 10,  xk = 1.861060942709,  f(xk) = 2.318724705799,  f'(xk) =
 -0.110550812403,  f''(xk) = 3.144823126679
```

Run Custom 1.1

```
o = custom(f, 1, 1.5, 2, 10);
fprintf("Custom:\n")
for row=1:size(o,1)
    fprintf("k = %i,\t xk = %0.12f,\t f(xk) = % 0.12f\n",o(row, :))
end
```

```
Custom:
k = 1,  xk = 2.000000000000,  f(xk) =  2.335412653811
k = 2,  xk = 1.979306971299,  f(xk) =  2.328684216801
k = 3,  xk = 1.881459171969,  f(xk) =  2.317129339417
k = 4,  xk = 1.896580738126,  f(xk) =  2.316810354179
k = 5,  xk = 1.895288900889,  f(xk) =  2.316808488868
k = 6,  xk = 1.895491823988,  f(xk) =  2.316808419798
k = 7,  xk = 1.895494223557,  f(xk) =  2.316808419788
k = 8,  xk = 1.895493592208,  f(xk) =  2.316808419789
k = 9,  xk = 1.895590080093,  f(xk) =  2.316808434826
k = 10,  xk = 1.895501987455,  f(xk) =  2.316808419886
k = 11,  xk = 1.895494500296,  f(xk) =  2.316808419788
```

Define the function we are optimizing 1.2

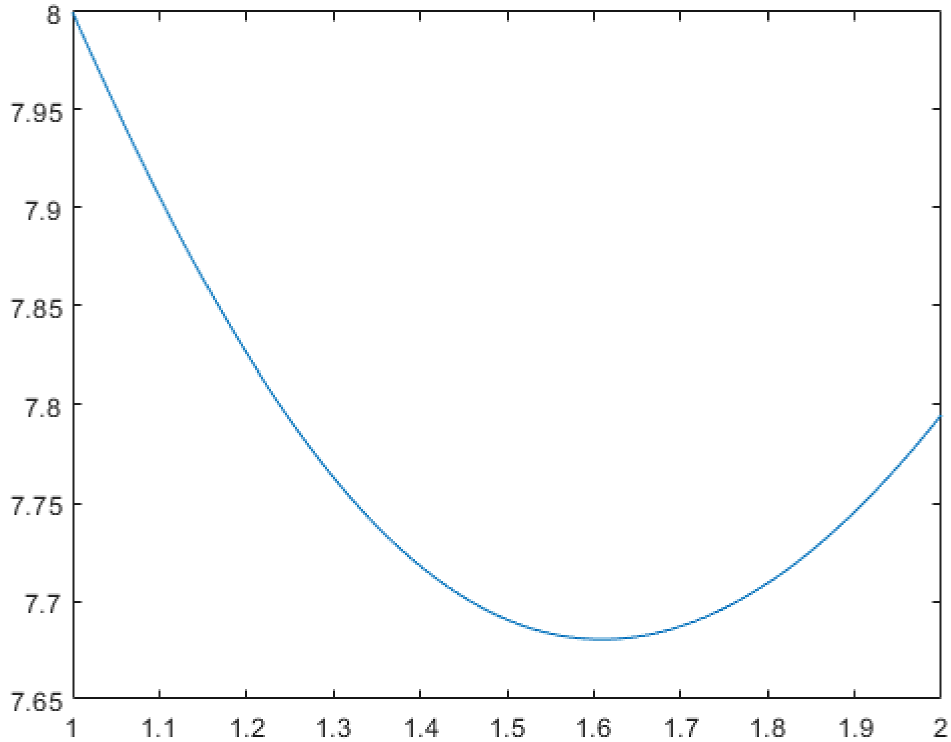
```
f = @(x) 8 * exp(1 - x) + 7 * log(x); % lambda function syntax
f1= @(x) -8 * exp(1-x) + 7 * 1/x;
f2= @(x) 8 * exp(1-x) - 7/ x.^2;
a = 1;
b = 2;

eps = 0.02;  % epsilon
```

Plot the function

```
x = a:0.01:b;
```

```
plot(x, f(x));
```



Run Golden Section 1.2

```
o = golden_sec(f, 1, 2, 0.02);
fprintf("Golden Section\n")
for row=1:size(o,1)
    fprintf("k = %i,\t ak = %0.12f,\t bk = %0.12f,\t f(ak) = %0.12f,\t
 f(bk) = %0.12f\t [%0.6f, %0.6f]\n",o(row, :))
end
```

```
Golden Section
k = 0,   ak = 1.000000000000,   bk = 2.000000000000,   f(ak) =
 8.000000000000,   f(bk) = 7.795065793291   [1.000000, 2.000000]
k = 1,   ak = 1.381966011250,   bk = 1.618033988750,   f(ak) =
 7.724695924128,   f(bk) = 7.795065793291   [1.381966, 2.000000]
k = 2,   ak = 1.618033988750,   bk = 1.763932022500,   f(ak) =
 7.724695924128,   f(bk) = 7.699467246802   [1.381966, 1.763932]
k = 3,   ak = 1.527864045000,   bk = 1.618033988750,   f(ak) =
 7.686003353484,   f(bk) = 7.699467246802   [1.527864, 1.763932]
k = 4,   ak = 1.618033988750,   bk = 1.673762078751,   f(ak) =
 7.686003353484,   f(bk) = 7.683813693632   [1.527864, 1.673762]
k = 5,   ak = 1.583592135001,   bk = 1.618033988750,   f(ak) =
 7.680996340185,   f(bk) = 7.683813693632   [1.583592, 1.673762]
k = 6,   ak = 1.618033988750,   bk = 1.639320225002,   f(ak) =
 7.680996340185,   f(bk) = 7.681179501562   [1.583592, 1.639320]
```

```
k = 7,  ak = 1.604878371253,  bk = 1.618033988750,  f(ak) =
 7.680996340185,  f(bk) = 7.680507437210  [1.583592, 1.618034]
k = 8,  ak = 1.596747752498,  bk = 1.604878371253,  f(ak) =
 7.680577644873,  f(bk) = 7.680507437210  [1.596748, 1.618034]
k = 9,  ak = 1.604878371253,  bk = 1.609903369994,  f(ak) =
 7.680462600136,  f(bk) = 7.680507437210  [1.604878, 1.618034]
```

Run Newton 1.2

```
o = newtons(f, f1, f2, 1, 10);
fprintf("Newtons:\n")
for row=1:size(o,1)
    fprintf("k = %i,\t xk = %0.12f,\t f(xk) = %0.12f,\t f'(xk) =
 %0.12f,\t f''(xk) = %0.12f\n",o(row, :))
end
```

```
Newtons:
k = 0,  xk = 1.000000000000,  f(xk) = 8.000000000000,  f'(xk) =
 -1.000000000000,  f''(xk) = 1.000000000000
k = 1,  xk = 2.000000000000,  f(xk) = 7.795065793291,  f'(xk) =
 0.556964470628,  f''(xk) = 1.193035529372
k = 2,  xk = 1.533153492150,  f(xk) = 7.685300898722,  f'(xk) =
 -0.128260959094,  f''(xk) = 1.715999639124
k = 3,  xk = 1.607897656524,  f(xk) = 7.680447702719,  f'(xk) =
 -0.002444130364,  f''(xk) = 1.648375432610
k = 4,  xk = 1.609380407550,  f(xk) = 7.680445890164,  f'(xk) =
 -0.000001087244,  f''(xk) = 1.646908206850
k = 5,  xk = 1.609381067723,  f(xk) = 7.680445890163,  f'(xk) =
 -0.000000000000,  f''(xk) = 1.646907552651
k = 6,  xk = 1.609381067723,  f(xk) = 7.680445890163,  f'(xk) =
 0.000000000000,  f''(xk) = 1.646907552651
k = 7,  xk = 1.609381067723,  f(xk) = 7.680445890163,  f'(xk) =
 0.000000000000,  f''(xk) = 1.646907552651
k = 8,  xk = 1.609381067723,  f(xk) = 7.680445890163,  f'(xk) =
 0.000000000000,  f''(xk) = 1.646907552651
k = 9,  xk = 1.609381067723,  f(xk) = 7.680445890163,  f'(xk) =
 0.000000000000,  f''(xk) = 1.646907552651
k = 10,  xk = 1.609381067723,  f(xk) = 7.680445890163,  f'(xk) =
 0.000000000000,  f''(xk) = 1.646907552651
```

Run Custom 1.2

```
o = custom(f, 1, 1.5, 2, 10);
fprintf("Custom:\n")
for row=1:size(o,1)
    fprintf("k = %i,\t xk = %0.12f,\t f(xk) = % 0.12f\n",o(row, :))
end
```

```
Custom:
k = 1,  xk = 2.000000000000,  f(xk) =  7.795065793291
k = 2,  xk = 1.623733494777,  f(xk) =  7.680615023576
k = 3,  xk = 1.613873622234,  f(xk) =  7.680462494961
k = 4,  xk = 1.608458688904,  f(xk) =  7.680446590873
```

```
k = 5,   xk = 1.609376284453,   f(xk) =  7.680445890182
k = 6,   xk = 1.609381486522,   f(xk) =  7.680445890164
k = 7,   xk = 1.609380785503,   f(xk) =  7.680445890163
k = 8,   xk = 1.609224213561,   f(xk) =  7.680445910424
k = 9,   xk = 1.609381792856,   f(xk) =  7.680445890164
k = 10,   xk = 1.609380628483,   f(xk) =  7.680445890164
k = 11,   xk = 1.609381906066,   f(xk) =  7.680445890164
```

Helper Function

```
function out = golden_sec(f, a, b, eps)
    k = 0;
    out = [k, a, b, f(a), f(b), a, b]; % array of output

    ro = (3 - sqrt(5))/2;

    while (b - a) >= eps
        k = k + 1;
        a1 = a + ro * (b - a);
        b1 = b - ro * (b - a);
        fa1 = f(a1);  % can store these to re-compute
        fb1 = f(b1);
        if fa1 > fb1
            a = a1;
        else
            b = b1;
        end
        out = [out ; [k, a1, b1, f(a), f(b), a, b]];
        %fprintf("k = %i,\t ak = %0.12f,\t bk = %0.12f,\t f(ak) =
 %0.12f,\t f(bk) = %0.12f\t [%0.6f, %0.6f]\n",k, a1, b1, fa1, fb1, a,
 b)
    end
end

function out = newtons(f, f1, f2, x, iter)
    k = 0;
    out = [k, x, f(x), f1(x), f2(x)];
    for i = 1:iter
        k = k + 1;
        x = x - f1(x)/f2(x);
        out = [out ; [k, x, f(x), f1(x), f2(x)]];
    end
end

function out = custom(f, x0, x1, x2, iter)
    k = 3;
    x = [x0, x1, x2];
    out = [k - 2, x(k), f(x(k))];
    for i = 1:iter
        s02 = x(k-0)^2 - x(k-2)^2;
        s10 = x(k-1)^2 - x(k-0)^2;
        s21 = x(k-2)^2 - x(k-1)^2;
        d12 = x(k-1) - x(k-2);
```

```
        d20 = x(k-2) - x(k-0);
        d01 = x(k-0) - x(k-1);

        num = -f(x(k-1))*s02 - f(x(k-2)) * s10 - f(x(k)) * s21;
        den = f(x(k)) * d12 + f(x(k-1)) * d20 + f(x(k-2)) * d01;
        x(k + 1) = num / (2 * den);
        k = k + 1;
        out = [out ; [k - 2, x(k), f(x(k))]];
    end
end
```

```
Golden Section
k = 0,  ak = 1.000000000000,  bk = 2.000000000000,  f(ak) =
 3.161209223473,  f(bk) = 2.335412653811  [1.000000, 2.000000]
k = 1,  ak = 1.381966011250,  bk = 1.618033988750,  f(ak) =
 2.660670580070,  f(bk) = 2.335412653811  [1.381966, 2.000000]
k = 2,  ak = 1.618033988750,  bk = 1.763932022500,  f(ak) =
 2.429153603732,  f(bk) = 2.335412653811  [1.618034, 2.000000]
k = 3,  ak = 1.763932022500,  bk = 1.854101966250,  f(ak) =
 2.343707268370,  f(bk) = 2.335412653811  [1.763932, 2.000000]
k = 4,  ak = 1.854101966250,  bk = 1.909830056251,  f(ak) =
 2.319569958647,  f(bk) = 2.335412653811  [1.854102, 2.000000]
k = 5,  ak = 1.909830056251,  bk = 1.944271909999,  f(ak) =
 2.319569958647,  f(bk) = 2.320778769582  [1.854102, 1.944272]
k = 6,  ak = 1.888543819998,  bk = 1.909830056251,  f(ak) =
 2.319569958647,  f(bk) = 2.317146921623  [1.854102, 1.909830]
k = 7,  ak = 1.875388202502,  bk = 1.888543819998,  f(ak) =
 2.317465461693,  f(bk) = 2.317146921623  [1.875388, 1.909830]
k = 8,  ak = 1.888543819998,  bk = 1.896674438754,  f(ak) =
 2.316887339366,  f(bk) = 2.317146921623  [1.888544, 1.909830]
k = 9,  ak = 1.896674438754,  bk = 1.901699437495,  f(ak) =
 2.316887339366,  f(bk) = 2.316871642186  [1.888544, 1.901699]
```

Published with MATLAB® R2021a

Problem 2 Derive a one-dimensional minimization algorithm based on quadratic fit that only requires objective function values (but no derivatives). Specifically, derive an algorithm that computes x_{k+1} based on x_k, x_{k-1}, x_{k-2}, and $f(x_k), f(x_{k-1}), f(x_{k-2})$. Hint: To simplify notation, used $\delta_{i,j} = x_{k-i} - x_{k-j}$ and $\sigma_{i,j} = (x_{k-i})^2 - (x_{k-j})^2$

Bonus: implement your algorithm as a MATLAB script and apply it to the numerical problems, above. Note that you will needthree points to initialize the algorithm.

We want to solve this linear equation

$$\begin{bmatrix} x_k^2 & x_k & 1 \\ x_{k-1}^2 & x_{k-1} & 1 \\ x_{k-2}^2 & x_{k-2} & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} f(x_k) \\ f(x_{k-1}) \\ f(x_{k-2}) \end{bmatrix}$$

We use cramer's rule to solve this.

$$D = \begin{vmatrix} x_k^2 & x_k & 1 \\ x_{k-1}^2 & x_{k-1} & 1 \\ x_{k-2}^2 & x_{k-2} & 1 \end{vmatrix} = x_k^2\delta_{1,2} + x_k\sigma_{2,1} + x_{k-1}x_{k-2}\delta_{1,2}$$

$$D_a = \begin{vmatrix} f(x_k) & x_k & 1 \\ f(x_{k-1}) & x_{k-1} & 1 \\ f(x_{k-2}) & x_{k-2} & 1 \end{vmatrix} = f(x_k)\delta_{1,2} + f(x_{k-1})\delta_{2,0} + f(x_{k-2})\delta_{0,1}$$

$$Db = f(x_{k-1})\sigma 0,2 + f(x_{k-2})\sigma 1,0 + f(x_k)\sigma 2,1$$

$$D_c = x_k^2(x_{k-1}f(x_{k-2}) - x_{k-2}f(x_{k-1})) - x_k(x_{k-1}^2 f(x_{k-2}) - x_{k-2}f(x_{k-1})) + f(x_k)(x_{k-1}^2 x_{k-2} - x_{k-2}^2 x_{k-1})$$

We know for a quadratic the minimum occurs at $-b/2a$. So we can now solve using Cramer's rule.

$$
\begin{aligned}
x_{k+1} &= \frac{-b}{2a} \\
&= \frac{-D_b}{D} * \frac{D}{2 * D_a} \\
&= \frac{-D_b}{2 * D_a} \\
&= \frac{-f(x_{k-1})\sigma_{0,2} - f(x_{k-2})\sigma_{1,0} - f(x_k)\sigma_{2,1}}{2 * (f(x_k)\delta_{1,2} + f(x_{k-1})\delta_{2,0} + f(x_{k-2})\delta_{0,1})}
\end{aligned}
$$