Due Date: 23rd September 2021
Instructions:

1. Answer ALL questions in the spaces allocated.
2. In this assignment, you are required to show all your working.
3. Your answers must be written in the spaces provided. You can adjust the spaces allocated for the answers if you need more space. You can type your answers if you wish.
4. *The lecturer maintains the right to call students in individually and ask them questions on the assignments. This may result in an adjustment of the final assignment grade.*
5. *Upload (i) Your R code (ii) Your Data and (3) A softcopy of your assignment on myelearning as a pdf. In Dropbox 1. DO NOT SUBMIT AS A SINGLE ZIP FILE with all the documents.*

1. QUESTION 1 (OPTIMIZATION – function with two variables)

   (a) Do some reading on how you can find the maximum and minimum of a function f(x,y). You may come across terms like saddles point, partial derivatives etc. After you have read that, find the maximum, minimum and saddle points (if any) exists for the following function:

$$f(x,y) = e^{-\frac{1}{3}x^2+x-y^3}$$

[10 marks]

In order to find critical points (maxima, minima and saddle) of a multivariate function, we find partial derivatives with respect to each individual variable. From the above after using chain rule for the derivation of nested functions:

$$f_x(x,y) = \left(-\frac{2}{3}x+1\right)e^{-\frac{1}{3}x^2+x-y^3}$$

$$f_y(x,y) = -3y^2e^{-\frac{1}{3}x^2-y^3}$$

Both expressions are set to zero:

$$f_x(x,y) = 0$$

$$\left(-\frac{2}{3}x+1\right)e^{-\frac{1}{3}x^2-y^3} = 0$$

Since e^m cannot be zero for all real values of m:

$$e^{-\frac{1}{3}x^2+x-y^3} \neq 0$$

$$\Rightarrow -\frac{2}{3}x+1 = 0$$

solving the above for x

$$x = \frac{3}{2}$$

Doing the same for $f_y(x,y)$:

$$f_y(x,y) = 0$$

$$-3y^2e^{-\frac{1}{3}x^2+x-y^3} = 0$$

Following from the above, e^m cannot be equal to zero for all real values of m:

$$\begin{aligned} e^{-\frac{1}{3}x^2+x-y^3} &\neq 0 \\ \Rightarrow -3y^2 &= 0 \\ y &= 0 \end{aligned}$$

From the above, $(\frac{3}{2}, 0)$ is a critical point in $f(x, y)$. To determine the nature of this critical point, we take the second derivative in both variables using chain rule:

$$\begin{aligned} f_{xx}(x,y) &= \left(-\frac{2}{3}x+1\right)^2 \cdot e^{-\frac{1}{3}x^2+x-y^3} + \left(-\frac{2}{3}\right)\left(e^{-\frac{1}{3}x^2+x-y^3}\right) \\ &= e^{-\frac{1}{3}x^2+x-y^3}\left[\left(-\frac{2}{3}x+1\right)^2 - \frac{2}{3}\right] \\ &= e^{-\frac{1}{3}x^2+x-y^3}\left[-\frac{4}{9}x^2 - \frac{4}{3}x + \frac{1}{3}\right] \end{aligned}$$

Similarly:

$$\begin{aligned} f_{yy}(x,y) &= (-3y^2)^2 \cdot e^{-\frac{1}{3}x^2+x-y^3} - 6y \cdot e^{-\frac{1}{3}x^2+x-y^3} \\ &= (9y^4 - 6y)e^{-\frac{1}{3}x^2+x-y^3} \end{aligned}$$

At critical point $(1.5, 0)$, $f_{xx}(x, y) = -5.645$ and $f_{yy}(x, y) = 0$. At this point, this may be inconclusive as $f_{yy}(x, y) = 0$ at $(1.5, 0)$. Hence we use the second partials test to check the nature of the critical point, where:

$$D = f_{xx}(x_0, y_0)f_{yy}(x_0, y_0) - (f_{xy}(x_0, y_0))^2$$

We consider four case:

(i) If $D > 0$ and $f_{xx}(x_0, y_0) > 0$, then f is concave up at this point and is therefore a local minimum

(ii) If $D > 0$ and $f_{xx}(x_0, y_0) < 0$, then f is concave down at this point and is therefore a local maximum.

(iii) If $D < 0$ then f has a saddle point at (x_0, y_0)

(iv) If $D = 0$, then this test is inconclusive

From the above, $f_x(x, y) = \left(-\frac{2}{3}x + 1\right)e^{-\frac{1}{3}x^2+x-y^3}$, finding $f_{x,y}(x, y)$:

$$f_{xy}(x,y) = -3y^2\left(-\frac{2}{3}x+1\right)e^{-\frac{1}{3}x^2+x-y^3}$$

Finding $f_{x,y}(x, y)$ at critical point $(1.5, 0)$: $f_{x,y}(1.5, 0) = 0$

$D = 0$, hence by means of the second partials test, this is inconclusive. In other terms, the second partials test cannot analytically deduce the exact nature of the critical point at $(x_0 = \frac{3}{2}, y_0 = 0)$

(b) Use R to plot a the function $f(x, y)$. Use the following range $-2 < x < 2$ and $-2 < y < 2$.

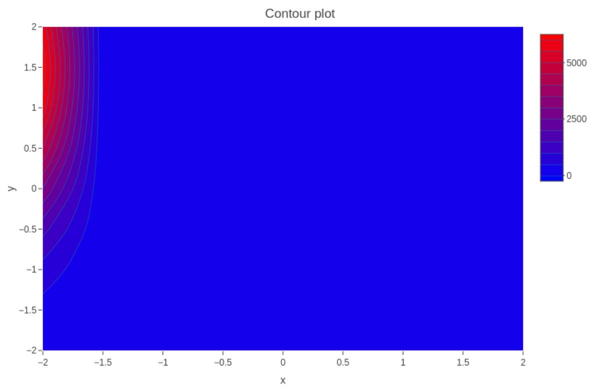
(i) Rcode

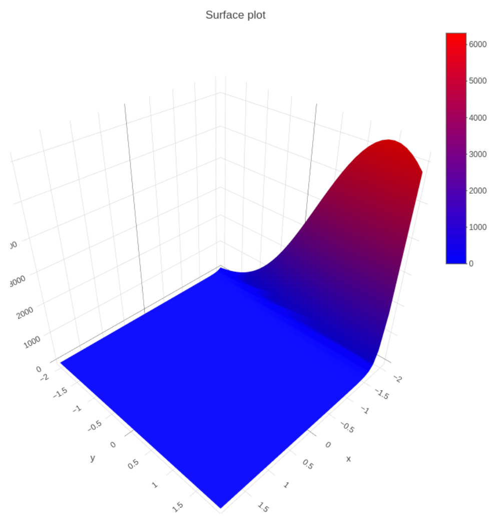
```
# Specify range of x and y values
xs <- seq(from = -2.0, to = 2.0, by = 0.5)
ys <- seq(from = -2.0, to = 2.0, by = 0.5)

#define the function
fx <- function (x, y) {
    return (exp(-(1/3)*(x^2) + x - y^3))
```

```
}
# find matrix of values in both directions and apply function
z <- outer(xs, ys, fx)

# plot graph
plot_ly(
    z = z, x=xs, y=ys,
    type="contour",
    colorscale='Bluered',
    width = 800, height = 500)
```

[2 marks]

(ii) Plot





[2 marks]

(c) Use the optim function in R to confirm your results in (a).

(i) R code & Output. The function need be redefined to accept input as a column vector in order for `optim` to work.

```
# redefine function so that optim() works
fx <- function (var) {
  return(exp(-(1/3)*var[1]^2 + var[1] - var[2]^3))
}

optim(
    c(0,0),
    fx,
    method='L-BFGS-B',
    lower=c(-2,-2),
    upper=c(2,2))

optim(c(0,0), fx, lower=c(-2,-2), upper=c(2,2))
```

```
$par
-2   1.3832524754988e-06

$value
0.0356739933472524
$counts
        function   12
        gradient   12

$convergence
0
$message
'CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH'
```

[2 marks]

2. Find the two parameter Weibull distribution in Wikipedia.

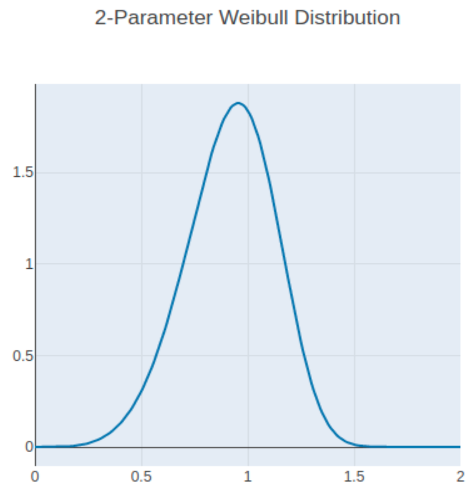
   (a) Write some R code to plot this distribution for the parameters $b = 5$ and $\eta = 1$.

      (i) R code

```r
library(ggplot2)
library(plotly)

weibull <- function(x, shape=1, scale=5) {
    (shape/scale)*(x/scale)^(shape-1)*exp(-(x/scale)^(shape))
}

x <- seq(0, 2, by=0.01)
f_x <- weibull(x, shape=5, scale=1)

#generate plot
data = data.frame(x, f_x)
fig <- plot_ly(
    data,
    x=x,
    y=f_x,
    name='2-param Weibull',
    type='scatter',
    mode='lines')

fig <- fig %>% layout(
    title = '2-Parameter Weibull Distribution',
    plot_bgcolor = "#e5ecf6")
fig <- fig %>% layout(
    autosize = F,
    width = 500,
    height = 500,
    margin = 0)

fig
```

[1 mark]

      (ii) Paste the plot here



[1 mark]

(iii) Use the newton's method in the course handout and write suitable functions for f, df and $df2$. Then generate 1,000 variates from a Weibull with $b = 5$ and $\eta = 1$ and check how well newton's method is able to recover the estimates of the parameters. You will follow the following steps to get this done:

A. code for generating the 1,000 variates from a Weibull with $b = 5$ and $\eta = 1$. Use `set.seed(1123)` to ensure everyone has the same data.

```
set.seed(1123)
shape <- 5
scale <- 1
n <- 1000
x <- rweibull(n, shape, scale)

# write data to output file for submission
write.table(f_x, file="weibull.txt", row.names=FALSE, col.names=FALSE)
```

[1 mark]

B. Find the score vector (column of first derivatives).

The PDF of the Weibull is given by:
$f(x|b,\eta) = \frac{b}{\eta}\left(\frac{x}{\eta}\right)^{b-1} e^{-\left(\frac{x}{\eta}\right)^b}$

The log-likelihood function is then found:
$f(\tilde{x}|b,\eta) = n[\ln b - b\ln\eta] + (b-1)\sum_{i=1}^{n}\ln X_i - \sum_{i=1}^{n}\left(\frac{X_i}{\eta}\right)$
To find the vector of score functions, the partial derivatives of the above need be found:

$$\frac{\delta\mathcal{L}}{\delta\eta} = -\frac{nb}{\eta} + \frac{\eta}{b}\sum_{i=1}^{n}\frac{X_i}{\eta}^{b}$$

$$\frac{\delta\mathcal{L}}{\delta b} = \frac{n}{b} - n\ln\eta + \sum_{i=1}^{n}\ln X_i - \sum_{i=1}^{n}\frac{X_i}{\eta}^{b}\ln\frac{X_i}{\eta}$$

[4 marks]

C. Write an R function for the first derivatives (df) which you will use in the Newton Raphson method.

```
df <- function(t) {
```

```
    eta <- t[1]
    b <- t[2]

    score = rep(0,2)
    score[1] = n/b-n*log(eta)+sum(log(x))-sum((x/eta)^(b)*log(x/eta))
    score[2] = -(n*b)/eta + (eta/b)*sum((x/eta)^b)

    return(score)
}
```

[4 marks]

D. Find the hessian matrix (matrix of second derivatives which we call $df2$). To find the Hessian matrix, the second partial derivatives with respect to both η and b need be found:

$$\frac{\delta^2\mathcal{L}}{\delta b^2} = \frac{\eta}{b^2}\left[n - (\eta - 1)\sum_{i=1}^{n}\frac{X_i}{b}^{\eta}\right]$$

$$\frac{\delta^2\mathcal{L}}{\delta \eta^2} = \frac{n}{\eta^2} - \sum_{i=1}^{n}\frac{X_i}{b}^{\eta}\left[\ln\frac{X_i}{b}\right]^2$$

$$\frac{\delta^2\mathcal{L}}{\delta b\delta\eta} = \frac{\delta^2\mathcal{L}}{\delta\eta\delta b} = -\frac{1}{b}\left[n - \sum_{i=1}^{n}\frac{X_i}{b}^{\eta} - \eta\sum_{i=1}^{n}\left(\frac{X_i}{b}\right)^{\eta}\ln\frac{X_i}{b}\right]$$

[4 marks]

E. Write an R function for the hessian matrix ($df2$) which you will use in the newton Raphson method.

```
df2 <- function(t) {
    eta <- t[1]
    b <- t[2]

    h <- matrix(0,2,2)
    h[1,1] <- (b/eta^2)*(n-(b-1)*sum((x/eta)^b))
    h[2,2] <- n/b^2 -sum((x/eta)^(b) * (log(x/eta))^2)
    h[1,2] <- -(1/eta)*(n-sum((x/eta)^b)-b*sum(log(x/eta)*(x/eta)^b))
    h[2,1] <- h[1,2]

    return(h)
}
```

[4 marks]

F. Apply the Newton Raphson Method in R and give the output. Comment on your results. Using the `newton` function provided in lecture slides, Newton-Rhapson in R was able to estimate the following values for scale $\hat{\eta} = 1.028$ and shape $\hat{b} = 4.950$. This is within margin of error for this method. I experimented with a lower value for the tolerance parameter where `tol=1e-10`, however this made no impact on the method's results [4 marks]