Gradient Descent

Objective: To optimize the given function using gradient descent algorithm $f(x) = 2.4\ (x-2)^2 + 3$ as part of Lab Migration Project.

Methods:

(i) Create sequence of elements in a vector.
(ii) Define the given function to optimize and then plot the function.
(iii) Calculate the gradient descent function and then iteratively find a solution.
(iv) Print final value of x and then plot the graph.
(v) Conclusion

```
rm(list=ls())
#Create a sequence of elements in a Vector
xs <- seq(0,4,len=20)
xs
```

```
##  [1] 0.0000000 0.2105263 0.4210526 0.6315789 0.8421053 1.0526316 1.2631579
##  [8] 1.4736842 1.6842105 1.8947368 2.1052632 2.3157895 2.5263158 2.7368421
## [15] 2.9473684 3.1578947 3.3684211 3.5789474 3.7894737 4.0000000
```

```
#Define the function to optimize
f <-  function(x) {2.4 * (x-2)^2 + 3}

#Plot the function
plot(xs,f(xs),type="l",xlab="x",ylab=expression(2.4(x-2)^2 + 3))

#Calculate the gradient (df/dx)
grad <- function(x){
  2.4*2*(x-2)
}
```

Inference: df/dx = 4.8(x-2), if x = 2 then 4.8(2-2) = 0. The actual solution we will approximate with gradient descent is x = 2.

```
#Gradient Descent Implementation
x <- 0.1  #initialize the first guess for x-value
xtrace <- x  #store x values for graphing purposes
ftrace <- f(x)  #store y values (function evaluated at x) for graphing
purposes
stepFactor <- 0.01  #learning rate 'alpha'
for (step in 1:5000) {
```

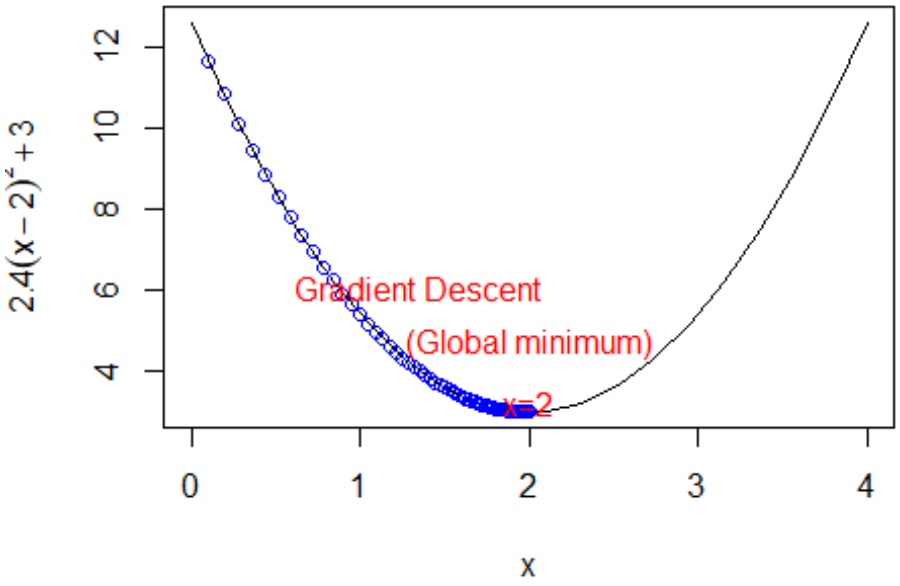
```
  x <- x - stepFactor*grad(x)  #gradient descent update
  xtrace <- c(xtrace,x)  #update for graph
  ftrace <- c(ftrace,f(x))  #update for graph
}

lines(xtrace,ftrace,type="b",col="blue")
text(0.5,6,"Gradient Descent",col="red",pos=4)

#Print final value of x
print(x)  #x converges to 2.0

## [1] 2

text(2,4,"x=2",col="red",pos=1)
text(2,4,"(Global minimum)",col="red",pos=3)
```



Inference: Now using plotting function we produce the plots, and populate these with points using the gradient descent algorithm. From the graph, the global minimum and descent of graph is clearly seen.

Conclusion:

Gradient descent is an iterative first-order optimization algorithm used to find a local minimum/maximum of a given function.

On applying gradient descent algorithm, to optimize the function $f(x) = 2.4\ (x-2)^2 + 3$, we found that the global minimum value, $x=2$.