Chapter 8: Performance Optimization

Brandon Morgan

1/17/2021

E9.7

We are given the following vector function in reduced form:

$$F(X) = (x_1 + x_2)^4 + 2(x_2 - 1)^2$$

The gradient of a function is given by:

$$\nabla F(X) = \begin{bmatrix} \frac{\partial}{\partial x_1} F(x) \\ \frac{\partial}{\partial x_2} F(x) \end{bmatrix}$$

which can be shown to be:

$$\nabla F(X) = \begin{bmatrix} 4(x_1 + x_2)^3 \\ 4\left(x_1 + x_2\right)^3 + 4\left(x_2 - 1\right) \end{bmatrix}$$

The Hessian matrix is given by:

$$\nabla^2 F(x) = \begin{bmatrix} \frac{\partial^2}{\partial x_1^2} F(x) & \frac{\partial^2}{\partial x_1 \partial x_2} F(x) \\ \frac{\partial^2}{\partial x_2 \partial x_1} F(x) & \frac{\partial^2}{\partial x_2^2} F(x) \end{bmatrix}$$

Where it can be shown from our example that

$$\frac{\partial^2}{\partial x_1^2} F(x) = 12(x_1 + x_2)^2$$

$$\frac{\partial^2}{\partial x_2^2} F(x) = 12\left(x_1 + x_2\right)^2 + 4$$

$$\frac{\partial^2}{\partial x_1 \partial x_2} F(x) = \frac{\partial^2}{\partial x_2 \partial x_1} F(x) = 12\left(x_1 + x_2\right)^2$$

therefore,

$$\nabla^2 F(x) = \begin{bmatrix} 12(x_1 + x_2)^2 & 12\left(x_1 + x_2\right)^2 \\ 12\left(x_1 + x_2\right)^2 & 12\left(x_1 + x_2\right)^2 + 4 \end{bmatrix}$$

1

A second-order Taylor series approximation for vectors is given by:

$$F(x) = F(x^*) + \nabla F(x)|_{x=x^*}(x - x^*) + \frac{1}{2}\nabla^2 F(x)|_{x=x^*}(x - x^*)^2$$

where x^* is the point about which we expand upon, given in the question as $x^* = [-1, 1]^T$. We get the following working:

Vector Function

$$F([-1, 1]^T) = 0$$

Gradient

$$\nabla F([-1, 1]^T) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Hessian

$$\nabla^2 F([-1, 1]^T) = \begin{bmatrix} 0 & 0 \\ 0 & 4 \end{bmatrix}$$

Combined

Together, we get the following second order Taylor approximation from the form $F(x) = F(x^*) + \nabla F(x^*)(x - x^*) + \frac{1}{2}(x - x^*)^T \nabla^2 F(x^*)(x - x^*)^T$, where $x^* = [0, 0]$:

$$F(x) = 0 + \begin{bmatrix} 0 & 0 \end{bmatrix}(X - \begin{bmatrix} -1 \\ 1 \end{bmatrix}) + \frac{1}{2}(X - \begin{bmatrix} -1 \\ 1 \end{bmatrix})^T \begin{bmatrix} 0 & 0 \\ 0 & 4 \end{bmatrix}(X - \begin{bmatrix} -1 \\ 1 \end{bmatrix})$$

Which reduces down to a hyperplane:

$$F(x) = 2(x_2 - 1)^2$$

2

The point $x_0 = [-1, 1]^T$ is a minimum first if it is a stationary point, a point that satisfies $\nabla F(X) = 0$. We already computed from part 1 during our derivation for the Taylor series approximation that

$$\nabla F([-1, 1]^T) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

, thus it is a stationary point, which is the first order condition. The second order condition evaluates the Hessian matrix at the point, as calculated earlier,

$$\nabla^2 F([-1, 1]^T) = \begin{bmatrix} 0 & 0 \\ 0 & 4 \end{bmatrix}$$

, where it can be shown that this matrix is only semi-positive definite, that is $z^T A z \geq 0$ for any vector $z \neq 0$, Eq. (8.30). Because our point satisfies the first order condition and our Hessian matrix evaluated at the point is positive semidefinite, the point is a strong or weak minimum.

3

Here we will perform one iteration of Newton's method from the initial guess $x_0 = [0.5, 0]^T$:

First, we need to evaluate our gradient and Hessian matrix at this value:

$$g_0 = \nabla F([0.5, 0]^T) = \begin{bmatrix} 0.0005 \\ -3.9995 \end{bmatrix}$$

$$A_0 = \nabla^2 F([0.5, 0]^T) = \begin{bmatrix} 0.03 & 0.03 \\ 0.03 & 4.03 \end{bmatrix}$$

Next, we will comput our next point x_1 by the following equation: $x_1 = x_0 - A_0^{-1} g_0$:

```r
A = matrix(c(0.03, 0.03, 0.03, 4.03), ncol=2)
g = matrix(c(0.0005, -3.9995), ncol=1)
init = matrix(c(0.5, 0), ncol=1)
init - solve(A)%*%g
```

```
##            [,1]
## [1,] -0.5166667
## [2,]  1.0000000
```

Plots

```r
library(purrr) # used for map2_dbl function
library(plotly) # used for 3D plotting
```

```
## Warning: package 'plotly' was built under R version 3.6.3

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 3.6.3

##
## Attaching package: 'plotly'

## The following object is masked from 'package:ggplot2':
##
##     last_plot

## The following object is masked from 'package:stats':
##
##     filter

## The following object is masked from 'package:graphics':
##
##     layout
```

```r
library(ggplot2) # used for plotting
```

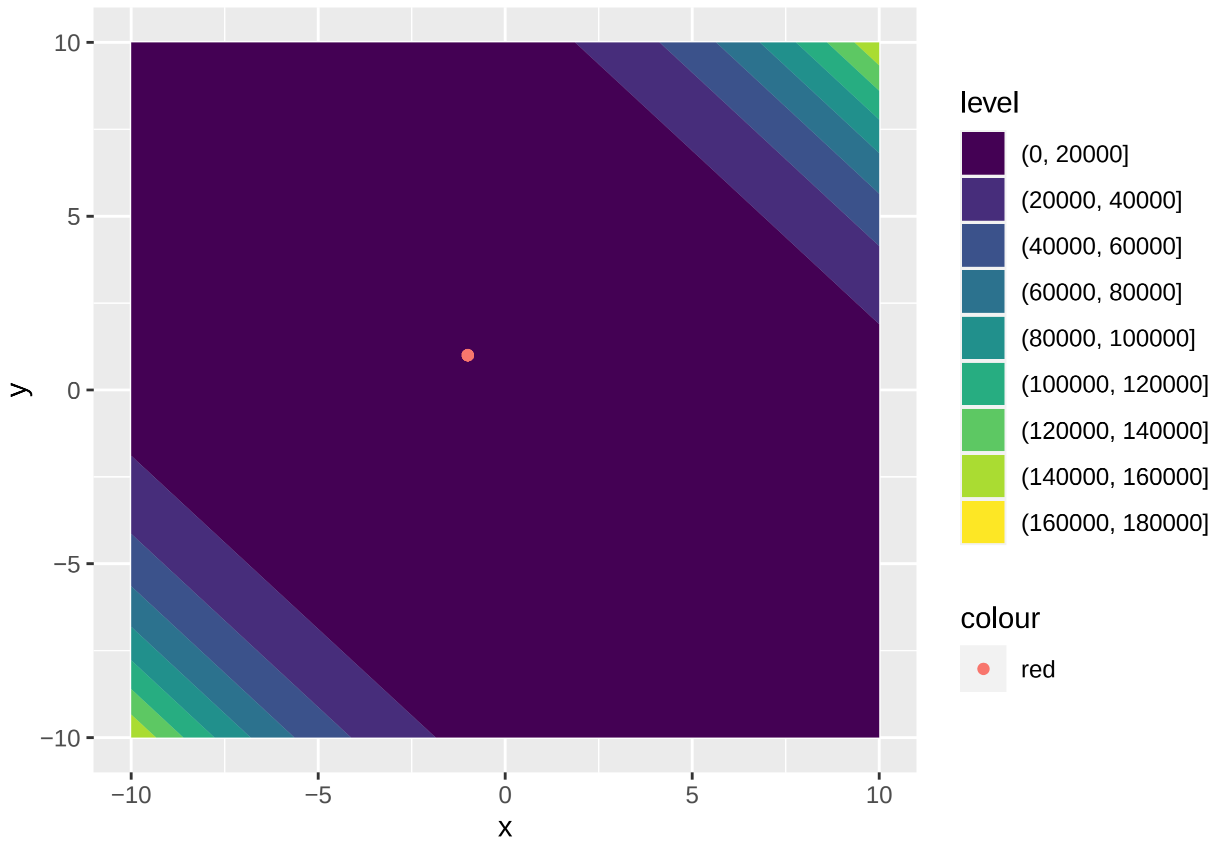
```r
fun = function(x, y) { # our original function
  (x+y)^4+2*(y-1)^2
}
```

```r
taylor = function(x, y) { # our taylor expansion
  2*(y-1)^2
  }
```

Original Function

Here we have a contour plot of our function along with the minimum point highlighted in red:

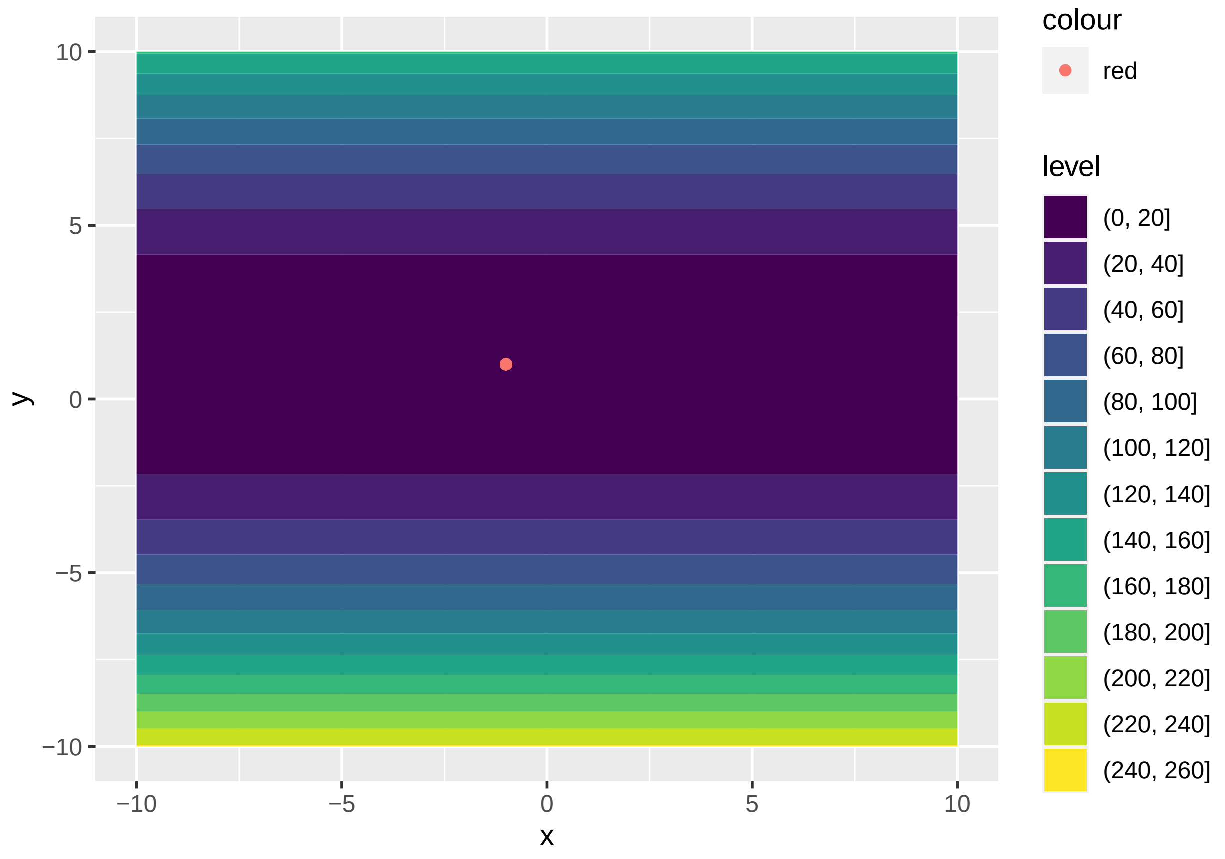
```r
# BOUNDS USED FOR CONTOUR
pointsX = seq(-10, 10, length=200)# create a 200x1 vector of values for x axis
pointsY = seq(-10, 10, length=200) # create a 200x1 vector of values for y axis
myGrid = expand.grid(pointsX, pointsY) # create 200x200 grid of points
z = map2_dbl(myGrid$Var1, myGrid$Var2, ~fun(.x, .y)) # maps all possible combinations
# of the grid through the function
x = myGrid$Var1
y =  myGrid$Var2
myPlot = plot_ly(x=~x, y=~y, z=~z, type="mesh3d", intensity=~z,
        colors = colorRamp(c("black","red","yellow","chartreuse3")))
myGrid$z = z
myPlot = add_trace(p=myPlot, z = c(fun(-1, 1)), x=c(-1), y=c(1), type="scatter3d")
# add another layer onto the 3D rendering, in this case a scatterplot
# of a single point, the stationary point, AKA minima
#myPlot
v = ggplot( myGrid,aes(x, y, z=z)) +geom_contour_filled()
v+geom_point(aes(x=-1, y=1, colour="red"))
```



Taylor Series

Here we have a contour plot of our second order Taylor series approximation about the point $x_0 = [-1, 1]$, which was shown to be either a strong or weak minimum.

```
# BOUNDS USED FOR CONTOUR
pointsX = seq(-10, 10, length=200)# create a 200x1 vector of values for x axis
pointsY = seq(-10, 10, length=200) # create a 200x1 vector of values for y axis
myGrid = expand.grid(pointsX, pointsY) # create 200x200 grid of points
z = map2_dbl(myGrid$Var1, myGrid$Var2, ~taylor(.x, .y)) # maps all possible combinations
# of the grid through the function
x = myGrid$Var1
y =  myGrid$Var2
myPlot = plot_ly(x=~x, y=~y, z=~z, type="mesh3d", intensity=~z,
        colors = colorRamp(c("black","red","yellow","chartreuse3")))
myGrid$z = z
myPlot = add_trace(p=myPlot, z = c(fun(-1, 1)), x=c(-1), y=c(1), type="scatter3d")
# add another layer onto the 3D rendering, in this case a scatterplot
# of a single point, the stationary point, AKA minima
#myPlot
v = ggplot( myGrid,aes(x, y, z=z)) +geom_contour_filled()
v+geom_point(aes(x=-1, y=1, colour="red"))
```


colour
red
level
(0, 20]
(20, 40]
(40, 60]
(60, 80]
(80, 100]
(100, 120]
(120, 140]
(140, 160]
(160, 180]
(180, 200]
(200, 220]
(220, 240]
(240, 260]
y
x
10
5
0
−5
−10
−10
−5
0
5
10

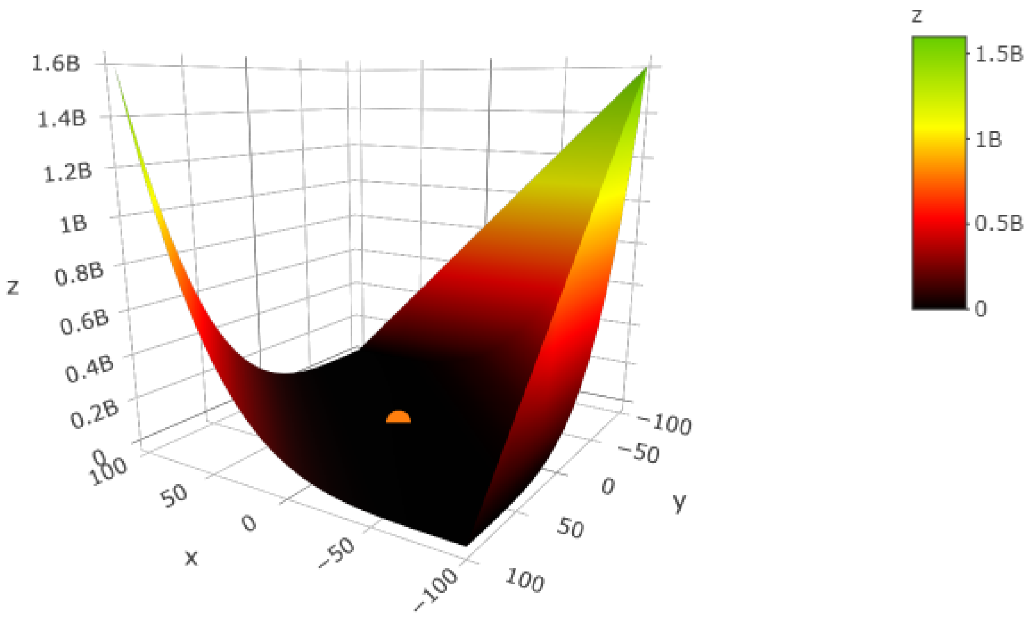


Figure 1: Original function with stationary point

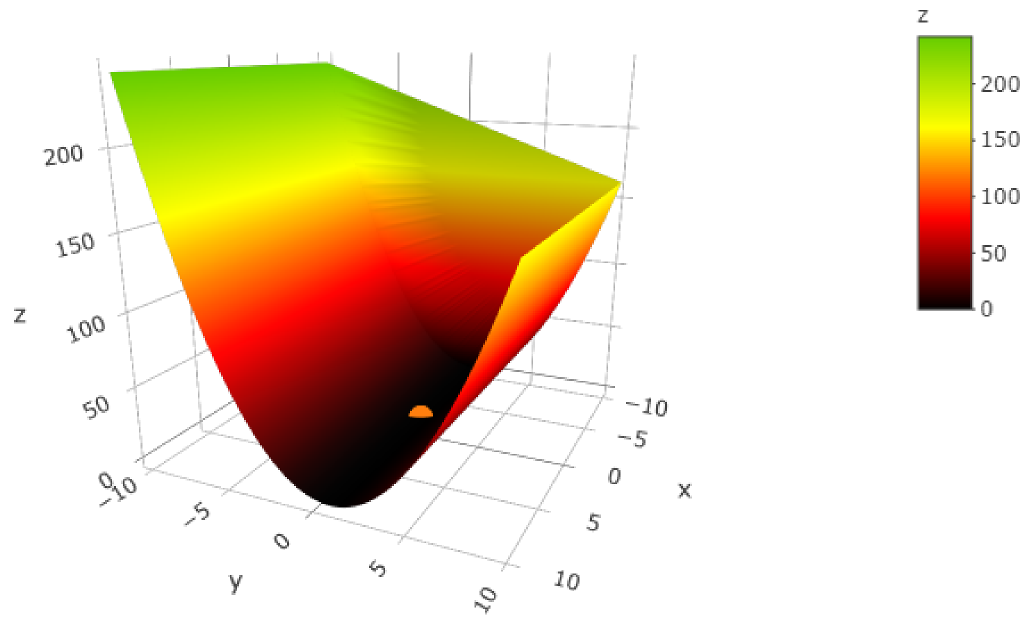


Figure 2: Taylor Series Expansion with stationary point