Homework 19

Ziyao Yang

4/7/23

1. [A] Show that f is convex

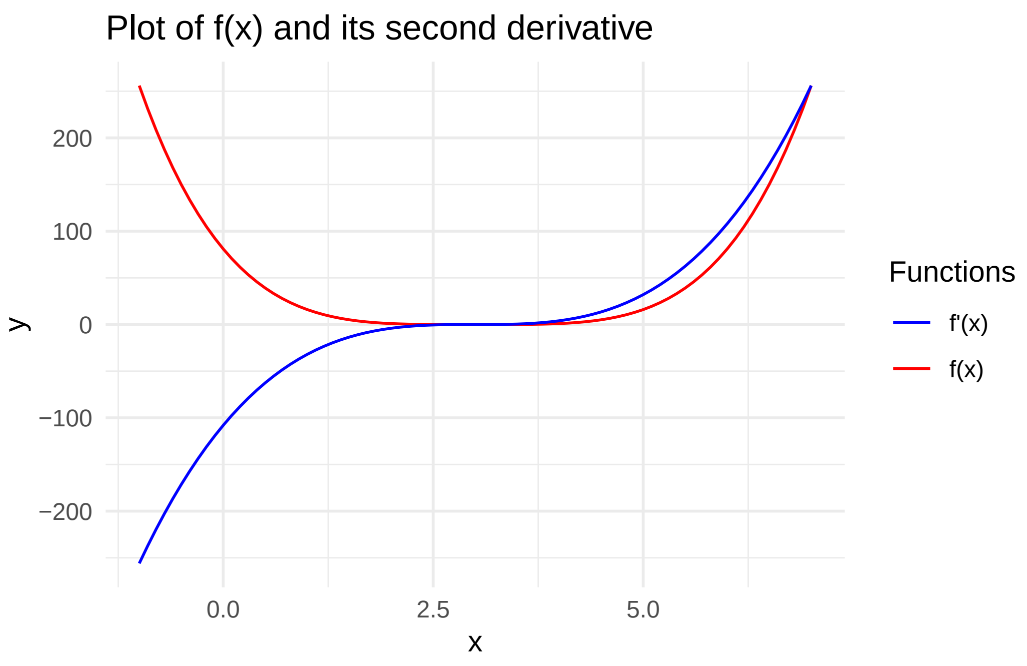
```r
f <- function(x) (x-3)^4
```

```r
# Load necessary libraries
library(ggplot2)

# Define the function f(x) and its derivatives
f <- function(x) (x - 3)^4
f_prime <- function(x) 4 * (x - 3)^3

# Define a range of x values to plot
x_vals <- seq(-1, 7, 0.1)

# Plot the function f(x) and its first derivative
ggplot(data.frame(x = x_vals), aes(x)) +
  geom_line(aes(y = f(x), color = "f(x)")) +
  geom_line(aes(y = f_prime(x), color = "f'(x)")) +
  scale_color_manual(values = c("blue", "red")) +
  labs(title = "Plot of f(x) and its second derivative",
       x = "x",
       y = "y",
       color = "Functions") +
  theme_minimal()
```



2. [A] Show that argminx Rf(x) = 3.

```r
# Use the optimize() function to find the argmin of f(x)
result <- optimize(f, interval = c(-10, 10))
# Display the result
cat("The argmin of f(x) is:", result$minimum)
```

```
The argmin of f(x) is: 3.00001
```

3. [A] Suppose you want to use Newton method to solve the optimization problem in the previous part, starting at x0 = 0. What is x1, the first value found by Newton method?

SEE ABOVE

4. [C] Write R codes to calculate x1 numerically to verify your analytic answer in the previous part.

```r
# Define the second derivatives of f(x)
f_double_prime <- function(x) 12 * (x - 3)^2

# Define the starting point x0
x0 <- 0

# Calculate x1 using the Newton-Raphson update formula
x1 <- x0 - (f_prime(x0) / f_double_prime(x0))
x1
```

```
[1] 1
```

5. [C] Write R codes to implement Newton method with the starting value x0 = 0 and a convergence tolerance of = 0.01. Compare your numerical solution with the theoretical solution argminx Rf(x) = 3.

```r
epsilon <- 0.01
```

```r
newtons_method_1d = function(start, d1, d2, tol) {
  prev_step = start
  new_step = start
  num_iters = 0
  while (TRUE) {
    # update Newton's method step
    new_step = prev_step - d1(prev_step) / d2(prev_step)
    num_iters = num_iters + 1
    # check for convergence
    if (abs(new_step - prev_step) <= tol) {
      print("Converged.")
      return(
        list(
          result=new_step,
          iters=num_iters
        )
      )
    } else {
      print(
        paste0("Iter: ", num_iters, ", Value: ", new_step)
      )
      prev_step = new_step
    }
  }
}
```

```
newton_res = newtons_method_1d(
  x0,
  f_prime,
  f_double_prime,
  tol = epsilon
)
```

```
[1] "Iter: 1, Value: 1"
[1] "Iter: 2, Value: 1.66666666666667"
[1] "Iter: 3, Value: 2.11111111111111"
[1] "Iter: 4, Value: 2.40740740740741"
[1] "Iter: 5, Value: 2.60493827160494"
[1] "Iter: 6, Value: 2.73662551440329"
[1] "Iter: 7, Value: 2.82441700960219"
[1] "Iter: 8, Value: 2.88294467306813"
[1] "Iter: 9, Value: 2.92196311537875"
[1] "Iter: 10, Value: 2.9479754102525"
[1] "Iter: 11, Value: 2.96531694016833"
[1] "Iter: 12, Value: 2.97687796011222"
[1] "Converged."
```

```
newton_res[[1]]
```

```
[1] 2.984585
```

```
3 - newton_res[[1]]
```

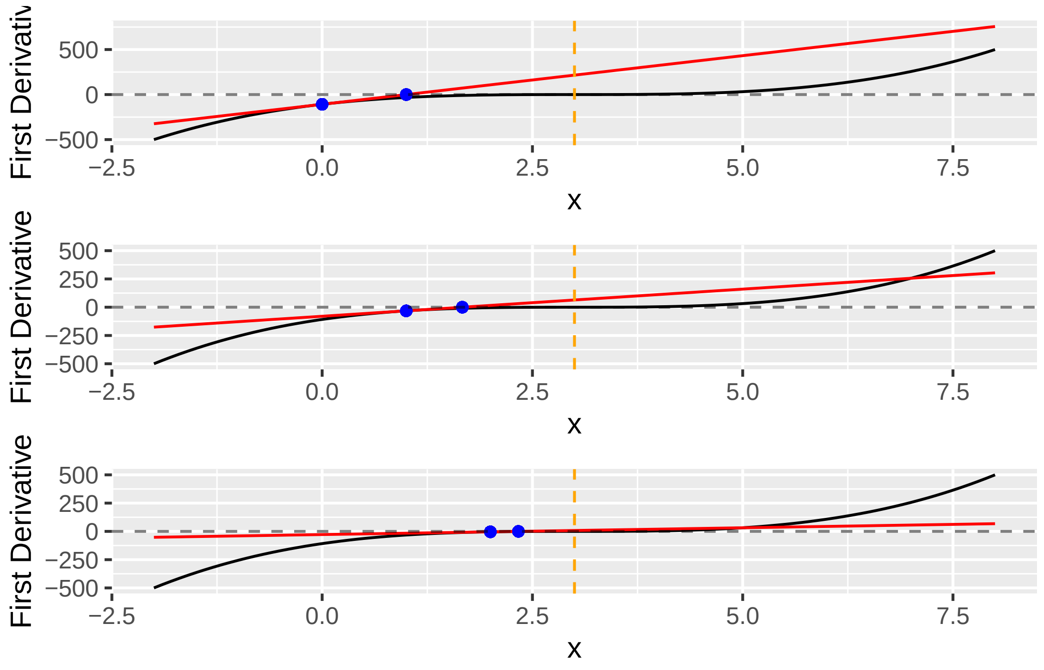
```
[1] 0.01541469
```

```
library(ggplot2)
library(gridExtra)

plot_newton_step = function(x_start) {
  x_vals <- seq(-2, 8, 0.1)

  ggplot(data = data.frame(x = x_vals,
                           y = f_prime(x_vals),
                           tan = f_prime(x_start) + (x_vals - x_start) * f_double_prime(x_star
    geom_line(aes(x = x, y = y)) +
    geom_line(aes(x = x, y = tan), color = "red") +
    geom_point(aes(x = x_start, y = f_prime(x_start)), color = "blue") +
    geom_point(aes(x = (x_start - f_prime(x_start) / f_double_prime(x_start)), y = 0), color =
    geom_hline(yintercept = 0, alpha = .5, linetype = "dashed") +
    geom_vline(xintercept = 3, color = "orange", linetype = "dashed") +
    xlab("x") +
    ylab("First Derivative")
}

grid.arrange(
  plot_newton_step(0),
  plot_newton_step(1),
  plot_newton_step(2)
)
```


First Derivativ
x
First Derivative
x
First Derivative
x