Lecture 2 Module 5: The Quadratic Equation

Exercises

Let's clear the global computing environment:

```
rm( list = ls() )
```

Exercises for Week 2 Module 5: The Quadratic Equation

Exercise 2.1: Quadratic Polynomials

Part (a)

Consider the quadratic function $f(x)$:

$$f(x) \ = \ 2x^2 - 4x - 16$$

First, determine the number of roots of this polynomial. Then, if there are any roots, calculate their value using the appropriate formula. Finally, create a graph so we can visualize the function and its roots.

Part (b)

Consider the quadratic function $g(x)$:

$$g(x) \ = \ 2x^2 - 4x + 8$$

First, determine the number of roots of this polynomial. Then, if there are any roots, calculate their value using the appropriate formula. Finally, create a graph so we can visualize the function and its roots.

Part (c)

Consider the quadratic function $h(x)$:

$$h(x) \ = \ 2x^2 - 4x + 2$$

First, determine the number of roots of this polynomial. Then, if there are any roots, calculate their value using the appropriate formula. Finally, create a graph so we can visualize the function and its roots.

Exercise 2.2 Newton's Method

So far, we've found one root of the quadratic polynomial $f(x) = x^2 - 6x + 5$. Now let's find the other, again by using Newton's method.

This time, start the procedure with an initial guess of 6. Run the process for 5 steps, reporting each intermediate result using a `cat()` statement, displaying the values with 6 decimal places.

Solution

Solutions to the Exercise

Exercise 2.1: Quadratic Polynomials

Part (a)

Consider the quadratic function $f(x)$:

$$f(x) \;=\; 2x^2 - 4x - 16$$

First, determine the number of roots of this polynomial. Then, if there are any roots, calculate their value using the appropriate formula. Finally, create a graph so we can visualize the function and its roots.

Solution

First, let's define variables for the coefficients of the quadratic function:

```
a <- 2

b <- -4

c <- -16
```

Then the discriminant is:

```
b^2 - 4 * a * c
```

```
## [1] 144
```

Since the discriminant is strictly greater than 0, there are exactly two roots for this polynomial

Let's calculate the first root:

```
first.root <- (-b - sqrt( b^2 - 4 * a * c ) ) / (2 * a)

cat(
    "The first root of the function is:",
    formatC(
        first.root,
        format = "f",
        digits = 2
    )
)
```

```
## The first root of the function is: -2.00
```

Let's check this:

```
a * first.root^2 + b * first.root + c
```

```
## [1] 0
```

Now let's calculate the second root, reporting our result using a `cat()` statement displaying 2 decimal places:

```
second.root <- (-b + sqrt( b^2 - 4 * a * c ) ) / (2 * a)

cat(
    "The second root of the equation is:",
    formatC(
        second.root,
        format = "f",
        digits = 2
    )
)
```

```
## The second root of the equation is: 4.00
```

Let's check this:

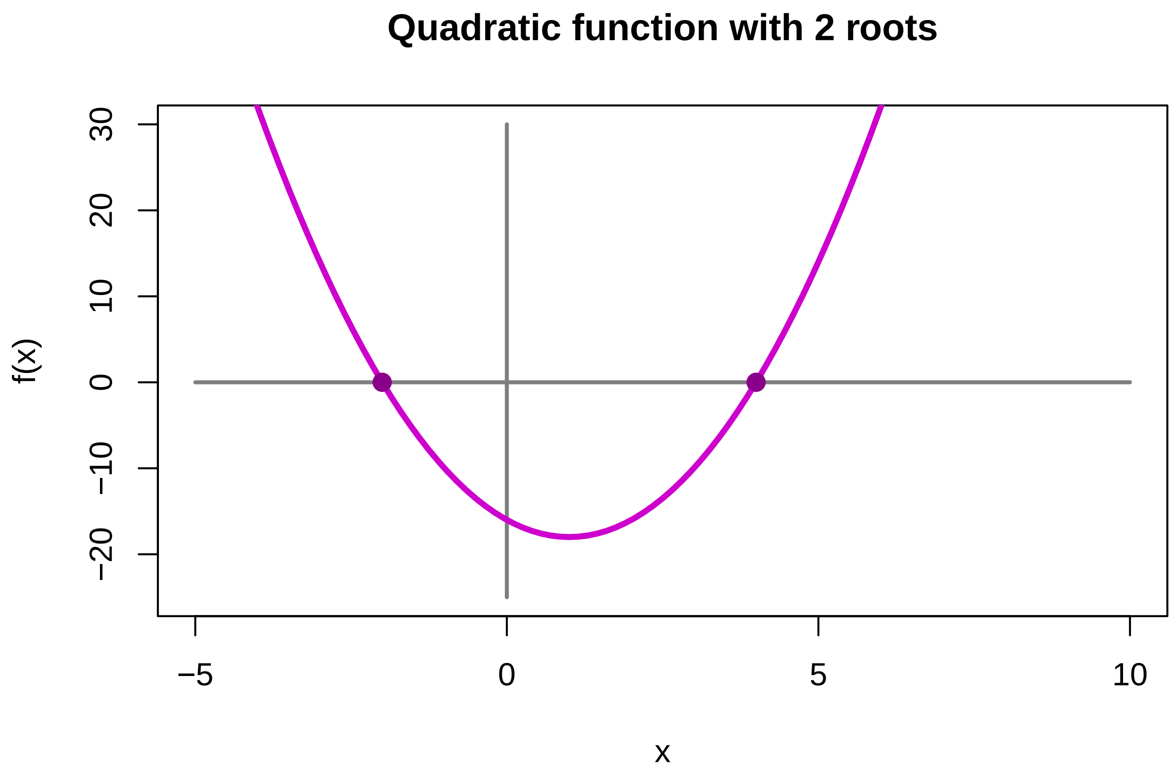
```
a * second.root^2 + b * second.root + c
```

```
## [1] 0
```

Now let's make a graph:

```
# First, let's create an empty plot:

plot(
    x = NULL,
    xlim = c(-5, 10),
    ylim = c(-25, 30),
    main = "Quadratic function with 2 roots",
    xlab = "x",
    ylab = "f(x)"
)


# Next, let's draw in the x- and y-axes:

segments(
    x0 = -5,
    y0 = 0,
    x1 = 10,
    y1 = 0,
    lty = "solid",
    lwd = 2,
    col = "gray50"
)


# Here's the vertical reference line:

segments(
    x0 = 0,
    y0 = -25,
    x1 = 0,
```

```
    y1 = 30,
    lty = "solid",
    lwd = 2,
    col = "gray50"
)


# Now we can graph the quadratic function:

curve(
    a*x^2 + b*x + c,
    lty = "solid",
    lwd = 3,
    col = "magenta3",
    add = TRUE
)


# Finally, we'll put points at the locations of the roots:

points(
    x = first.root,
    y = 0,
    pch = 19,
    cex = 1.2,
    col = "magenta4"
)

points(
    x = second.root,
    y = 0,
    pch = 19,
    cex = 1.2,
    col = "magenta4"
)
```



Part (b)

Consider the quadratic function $g(x)$:

$$g(x) \;=\; 2x^2 - 4x + 8$$

First, determine the number of roots of this polynomial. Then, if there are any roots, calculate their value using the appropriate formula. Finally, create a graph so we can visualize the function and its roots.

Solution

First, let's define variables for the coefficients of the quadratic function:

```
a <- 2

b <- -4

c <- 8
```

Then the discriminant is:

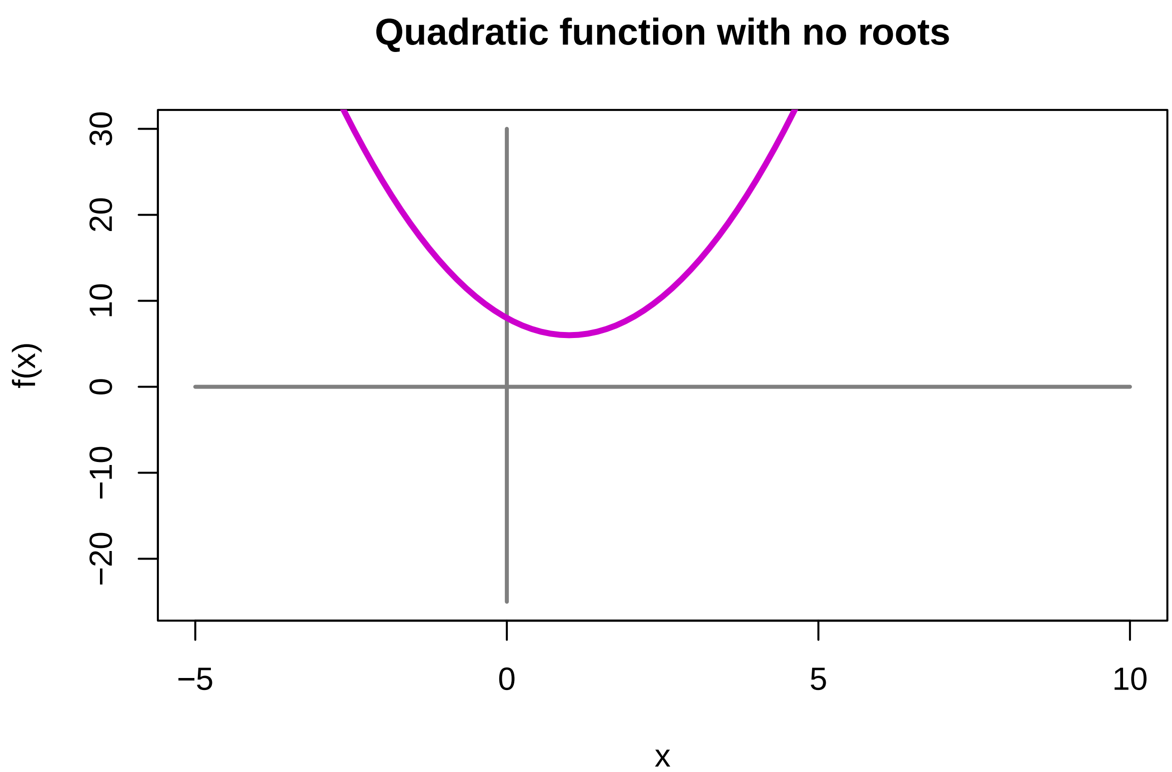
```
b^2 - 4 * a * c
```

```
## [1] -48
```

Since the discriminant is negative, there are no roots for this quadratic polynomial.

Now let's make a graph:

```r
# First, let's create an empty plot:

plot(
    x = NULL,
    xlim = c(-5, 10),
    ylim = c(-25, 30),
    main = "Quadratic function with no roots",
    xlab = "x",
    ylab = "f(x)"
)


# Next, let's draw in the x- and y-axes:

segments(
    x0 = -5,
    y0 = 0,
    x1 = 10,
    y1 = 0,
    lty = "solid",
    lwd = 2,
    col = "gray50"
)


# Here's the vertical reference line:

segments(
    x0 = 0,
    y0 = -25,
    x1 = 0,
    y1 = 30,
    lty = "solid",
    lwd = 2,
    col = "gray50"
)


# Now we can graph the quadratic function:

curve(
    a*x^2 + b*x + c,
    lty = "solid",
    lwd = 3,
    col = "magenta3",
    add = TRUE
)
```




```r
# Finally, there are no roots for this quadratic polynomial,
# so we're done!
```

Part (c)

Consider the quadratic function $h(x)$:

$$h(x) \ = \ 2x^2 - 4x + 2$$

First, determine the number of roots of this polynomial. Then, if there are any roots, calculate their value using the appropriate formula. Finally, create a graph so we can visualize the function and its roots.

Solution

First, let's define variables for the coefficients of the quadratic function:

```r
a <- 2

b <- -4

c <- 2
```

Then the discriminant is:

```r
b^2 - 4 * a * c
```

```
## [1] 0
```

Since the discriminant is zero, there is exactly one root for this quadratic polynomial.

```
only.root <-
    - b / (2 * a)

cat(
    "Only root of h(x):",
    formatC(
        only.root,
        format = "f",
        digits = 2
    )
)
```

```
## Only root of h(x): 1.00
```

Let's check this:

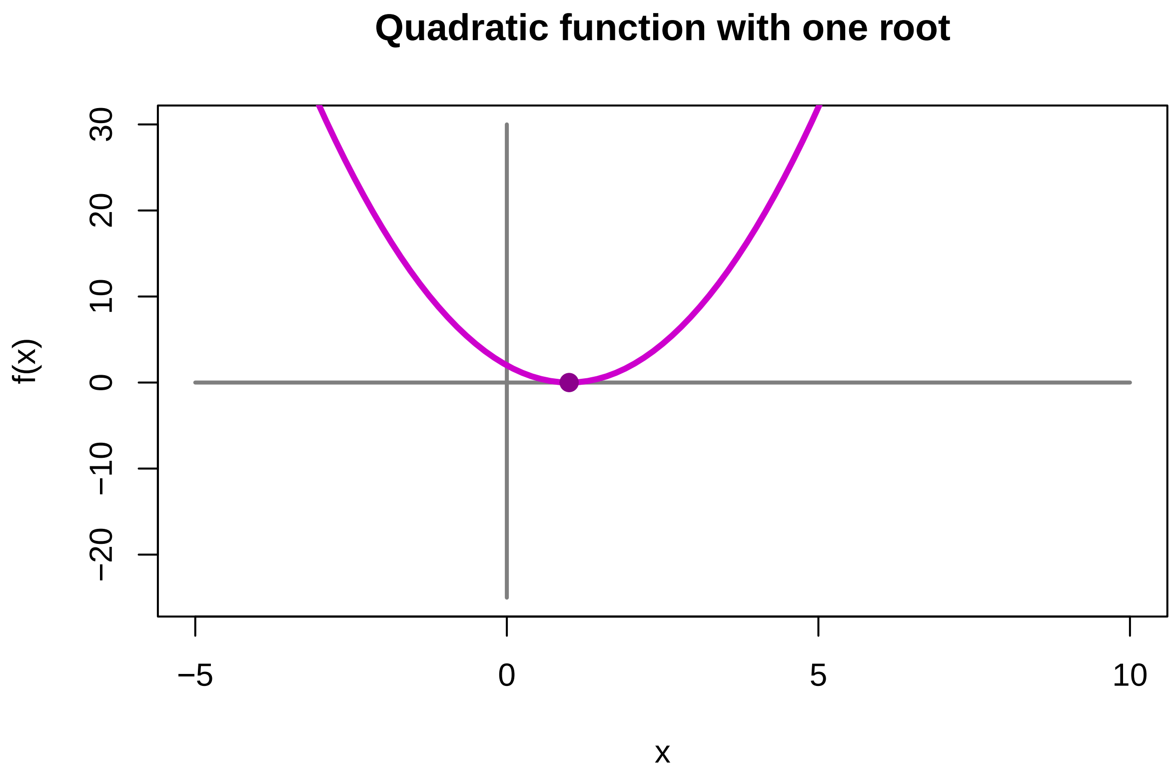
```
a * only.root^2 + b * only.root + c
```

```
## [1] 0
```

Now let's make a graph:

```
# First, let's create an empty plot:

plot(
    x = NULL,
    xlim = c(-5, 10),
    ylim = c(-25, 30),
    main = "Quadratic function with one root",
    xlab = "x",
    ylab = "f(x)"
)


# Next, let's draw in the x- and y-axes:

segments(
    x0 = -5,
    y0 = 0,
    x1 = 10,
    y1 = 0,
    lty = "solid",
    lwd = 2,
    col = "gray50"
)


# Here's the vertical reference line:

segments(
```

```r
    x0 = 0,
    y0 = -25,
    x1 = 0,
    y1 = 30,
    lty = "solid",
    lwd = 2,
    col = "gray50"
)


# Now we can graph the quadratic function:

curve(
    a*x^2 + b*x + c,
    lty = "solid",
    lwd = 3,
    col = "magenta3",
    add = TRUE
)


# Finally, there is only one root for this polynomial:

points(
    x = only.root,
    y = 0,
    pch = 19,
    cex = 1.2,
    col = "magenta4"
)
```



Exercise 2.2 Newton's Method

So far, we've found one root of the quadratic polynomial $f(x) = x^2 - 6x + 5$. Now let's find the other, again by using Newton's method.

This time, start the procedure with an initial guess of 6. Run the process for 5 steps, reporting each intermediate result using a `cat()` statement, displaying the values with 6 decimal places.

Solution

First, let's create some variables to hold the values of the coefficients:

```
a <- 1

b <- -6

c <- 5
```

Now let's make our initial guess:

```
x.1 <- 6
```

Now we'll calculate the second guess, and the report it:

```
x.2 <-
    (a * x.1^2 - c) /
    (2 * a * x.1 + b)

cat(
    "x.2 =",
    formatC(
        x.2,
        format = "f",
        digits = 6
    )
)
```

```
## x.2 = 5.166667
```

Now let's calculate and report our third guess:

```
x.3 <-
    (a * x.2^2 - c) /
    (2 * a * x.2 + b)

cat(
    "x.3 =",
    formatC(
        x.3,
        format = "f",
        digits = 6
    )
)
```

```
## x.3 = 5.006410
```

Now for the fourth estimate:

```
x.4 <-
    (a * x.3^2 - c) /
    (2 * a * x.3 + b)

cat(
    "x.4 =",
    formatC(
        x.4,
        format = "f",
        digits = 6
    )
)
```

```
## x.4 = 5.000010
```

And finally the fifth estimate:

```r
x.5 <-
    (a * x.4^2 - c) /
    (2 * a * x.4 + b)

cat(
    "x.5 =",
    formatC(
        x.5,
        format = "f",
        digits = 6
    )
)
```

```
## x.5 = 5.000000
```