Week 1 Module 2: Graphics Plots

CSCI E-5a: Programming in R

Let's clear the global computing environment:

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

Module Preview

Hello! And welcome to Module 2: Graphics Plots.

In this module, we'll learn about the basic features of graphics plots.

- In Section 1, we'll create a basic plot with a single point.
- In Section 2, we'll learn how to modify aspects of the basic plot by using options.
- In Section 3, we'll see how to modify aspects of the basic plot by using graphical parameters.
- In section 4, we'll learn how to create empty displays.

When you've completed this module, you'll be able to:

- Create a basic plot with a single point.
- Control the range of the x- and y-axes.
- Specify the text of the main title.
- Modify the text of the x- and y-axis titles.
- Suppress the display of the outside box.
- Specify a square display.
- Create an empty plot with no data.
- Create a completely blank display.

In this module, we'll meet 2 new built-in R functions:

- `plot()`
- `par()`

All right, let's start out by creating our first plot!

Section 1: Plotting a Single Point

> **Main idea** *We can draw a plot with a single point.*

In this section, we'll create a basic plot with a single point.

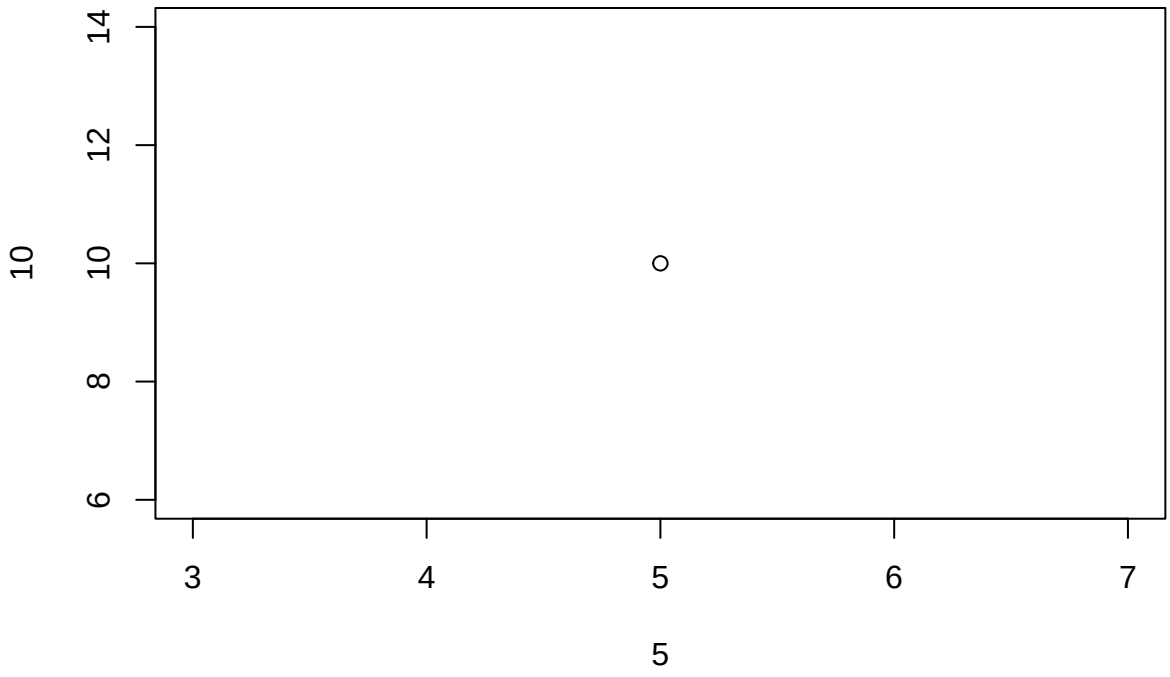
Let's start by creating the simplest graph possible.

We'll draw a single point at the location $x = 5$ and $y = 10$, and to do this we'll use the built-in R function `plot()`.

The `plot()` function takes two input arguments:

- The numeric value of the x coordinate.
- The numeric value of the y coordinate.

```
plot(
    x = 5,
    y = 10
)
```



Amazing! Our very first graph! A single dot!

Let's review what just happened:

- We *called* the function `plot()`, with two input arguments.

- The first input argument was named `x`, and we assigned the value 5 to this named argument.
- The second input argument was named `y`, and we assigned the value 10 to this named argument.
- The *plot()* function then constructed the graph, and displayed it.

Notice the box around the graph.

This box acts as a window, so that objects are only visible if they are inside the window.

All the space outside of the box is essentially a frame for the window.

If an R object is outside the range of the window, then it won't be observable.

In this model, the graphical space itself is effectively infinite, but we can only observe the finite region inside the plotting window, which is outlined by the box.

Let's take a moment to define some terms:

- The x-axis is the structure at the bottom of the graph, and consists of a horizontal line, small vertical tick marks, and labels for each of the ticks.
- The x-axis title is the text beneath the x-axis.
- The y-axis is the structure on the left-hand side of the graph, and consists of the vertical axis line, small horizontal tick marks, and labels for each of the ticks.
- The y-axis title is the text to the left of the y-axis.
- Although you can't see it in this graph, there is also a main title, which is located in the center above the graph.

You might not realize it at first, but a lot of things happened in making this very simple graph:

- First, R had to determine the range of x values for the x-axis.
- Next, R determined the range of y values for the y axis.
- R then created an x-axis and a y-axis, and labeled this with numbers.
- R then drew the box around the plotting region.
- R then created a title for the x axis and a title for the y axis.
- Finally, R drew the point as a small circle.

To be honest, this is actually a very primitive graph, and it is unacceptable as a finished image intended for presentation.

For the rest of this module we'll learn how to modify this display to improve it.

So that's how to create a basic plot with a single point.

Now let's see how to modify this display by using options.

Section 2: Options

> **Main idea** *We can modify aspects of the basic plot by using options*

In this section, we'll learn how to modify aspects of the basic plot by using options.

Now we'll learn how to customize some features of our simple plot.

One way to change the display is to use *options*.

In general, an "option" is something that you can choose to do, but don't have to do.

In R, an option in a function is something that you can specify if you want to, but don't have to.

The `plot()` function has many options, but for right now we'll focus on 6 fundamental ones:

- `xlim`
- `ylim`
- `main`
- `xlab`
- `ylab`
- `las`

Controlling the range of the x- and y-axes

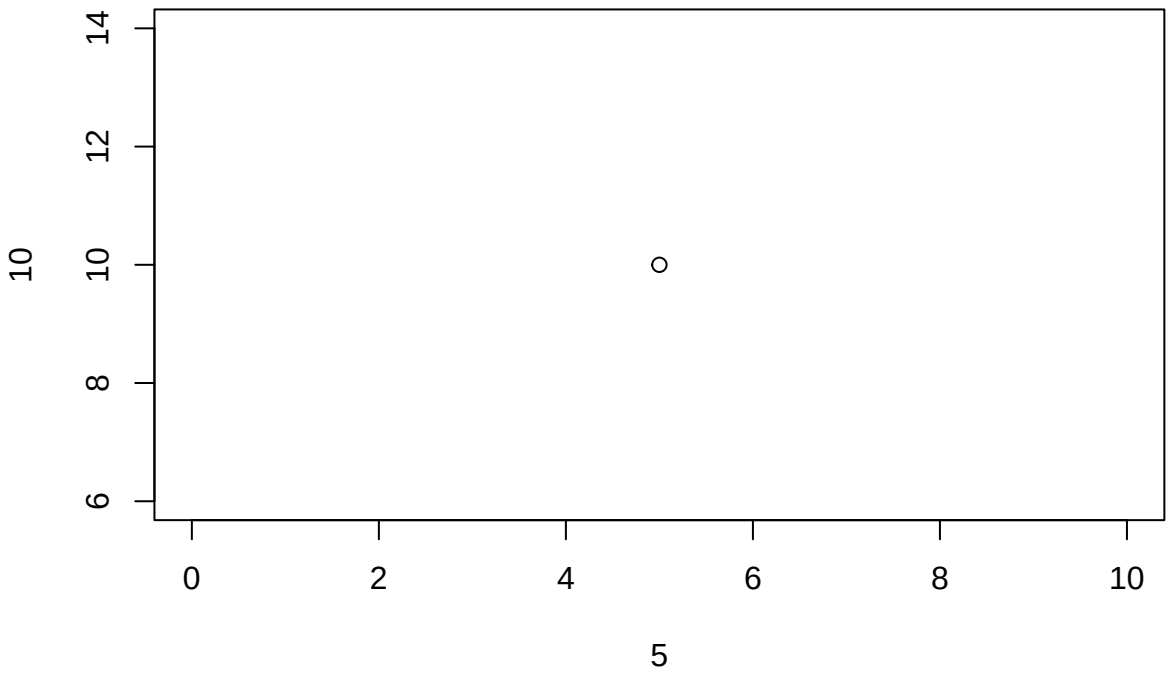
Notice that in our simple graph R automatically determines that the range of the x-axis will be from 3 to 7, and the range of the y-axis will be from 6 to 14.

We can also adjust these ranges.

To specify the range of the x-axis, use the `xlim` option along with a vector consisting of two elements: the first element determines the minimum x value, and the second element determines the maximum x value for the plotting region.

Let's re-do our basic graph, but this time we'll explicitly specify that the x-axis should range from 0 to 10:

```
plot(
    x = 5,
    y = 10,
    xlim = c(0, 10),  # Specifying the range of the x-axis
)
```

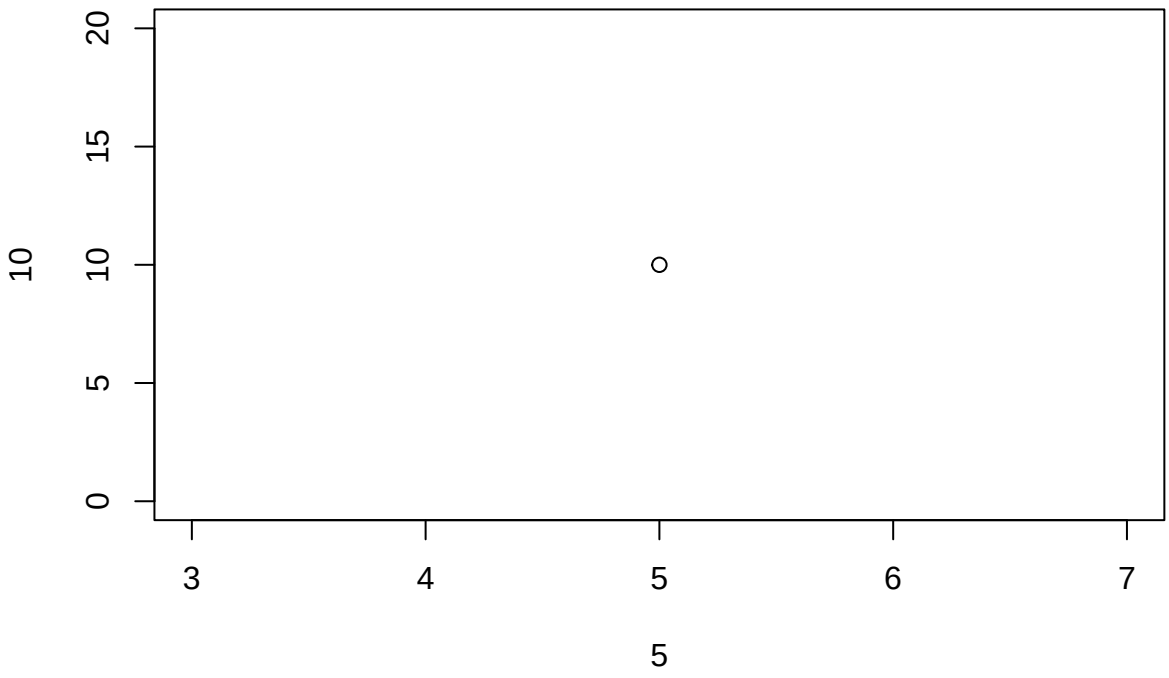


Make sure that you understand exactly how this graph differs from the original graph, and be able to explain how this was achieved.

Similarly, to specify the range of the y-axis, use the `ylim` option along with a vector consisting of two elements: the first element determines the minimum y value, and the second element determines the maximum y value for the plotting region.

Let's re-do our basic graph, but this time we'll explicitly specify that the y-axis should range from 0 to 20:

```
plot(
    x = 5,
    y = 10,
    ylim = c(0, 20)  # Specifying the range of the y-axis
)
```



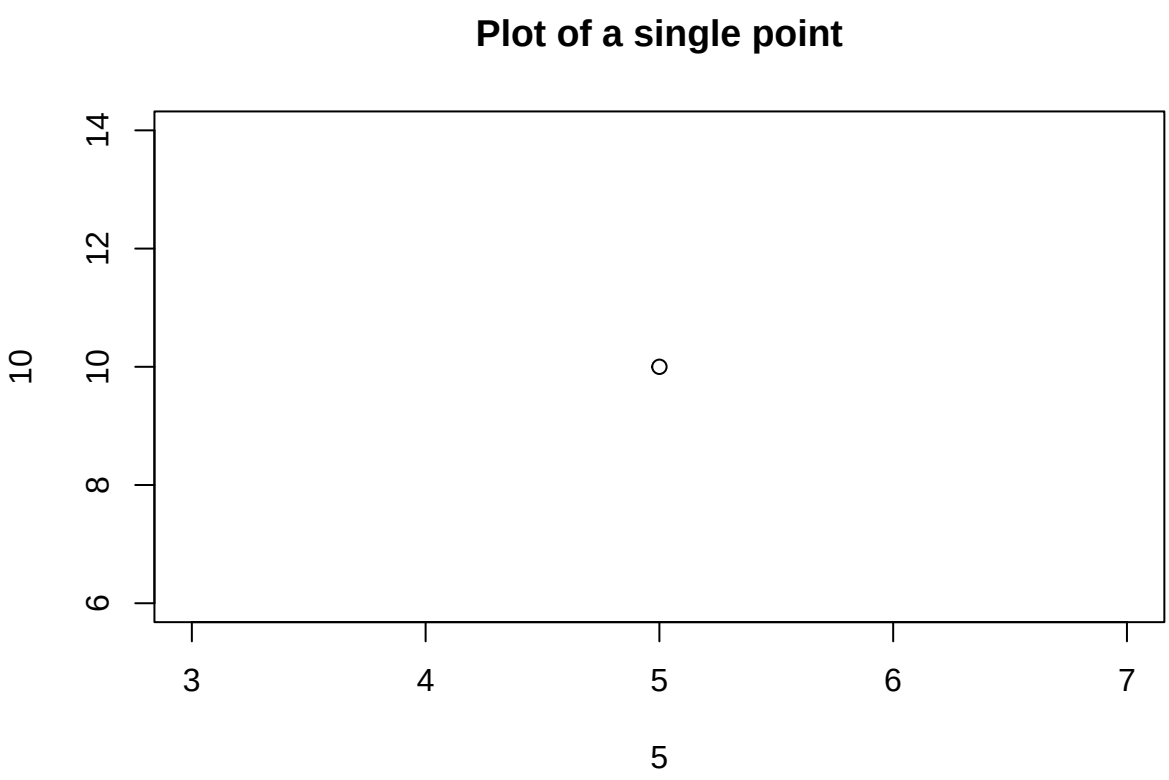
Adding a main title

Let's add a main title to this graph.

The main title will print in bold type face above the plotting region, and will be centered.

To specify the main title, we use the `main` option with a character string value:

```
plot(
    x = 5,
    y = 10,
    main = "Plot of a single point"   # Adding the main title
)
```

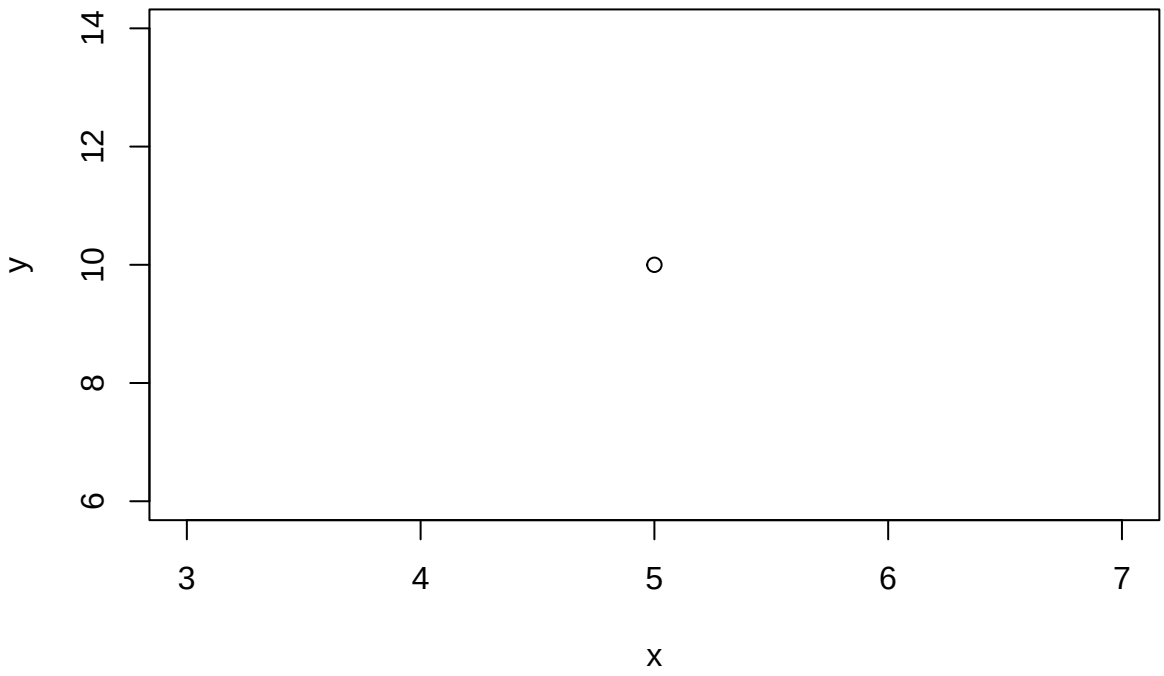


Make sure that you understand exactly how this graph differs from the original graph, and be able to explain how this was achieved.

Adjusting the x-axis and y-axis titles

Next, we can also adjust the titles of the x and y axes by using the `xlab` and `ylab` options:

```
plot(
    x = 5,
    y = 10,
    xlab = "x",  # Adding the x-axis title
    ylab = "y"   # Adding the y-axis title
)
```

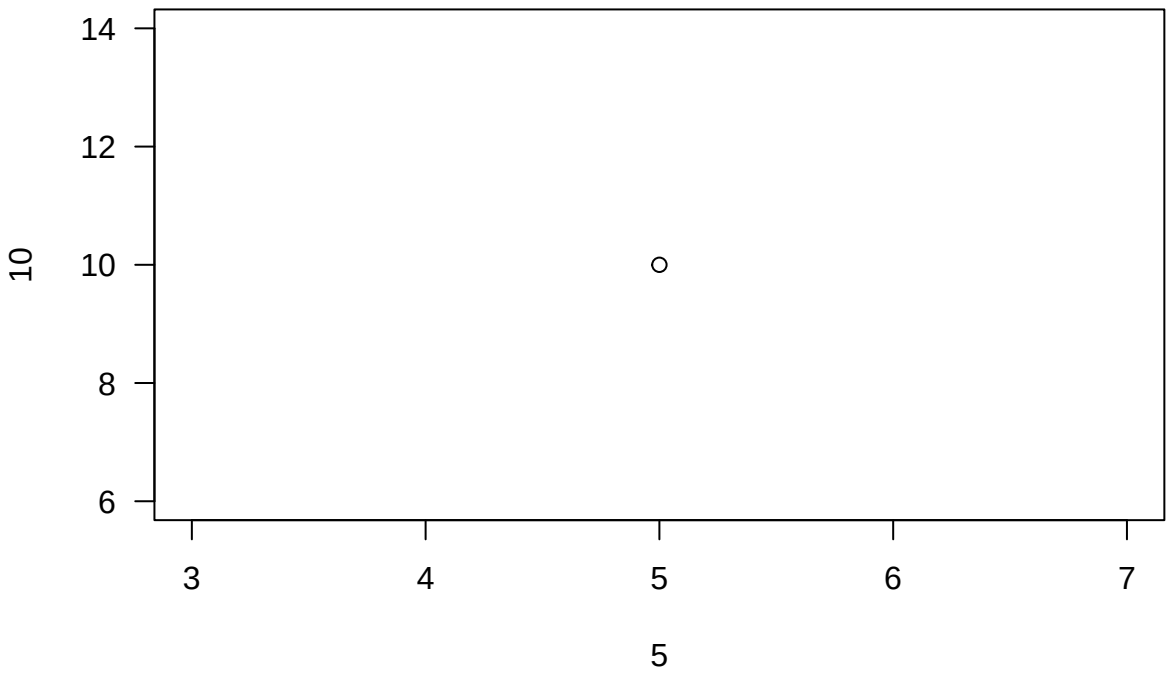


Make sure that you understand exactly how this graph differs from the original graph, and be able to explain how this was achieved.

Aligning the axis labels

If we use the `las = 1` option, the labels on the y axis will be displayed vertically:

```
plot(
    x = 5,
    y = 10,
    las = 1  # Specifying the alignment of the axis labels
)
```

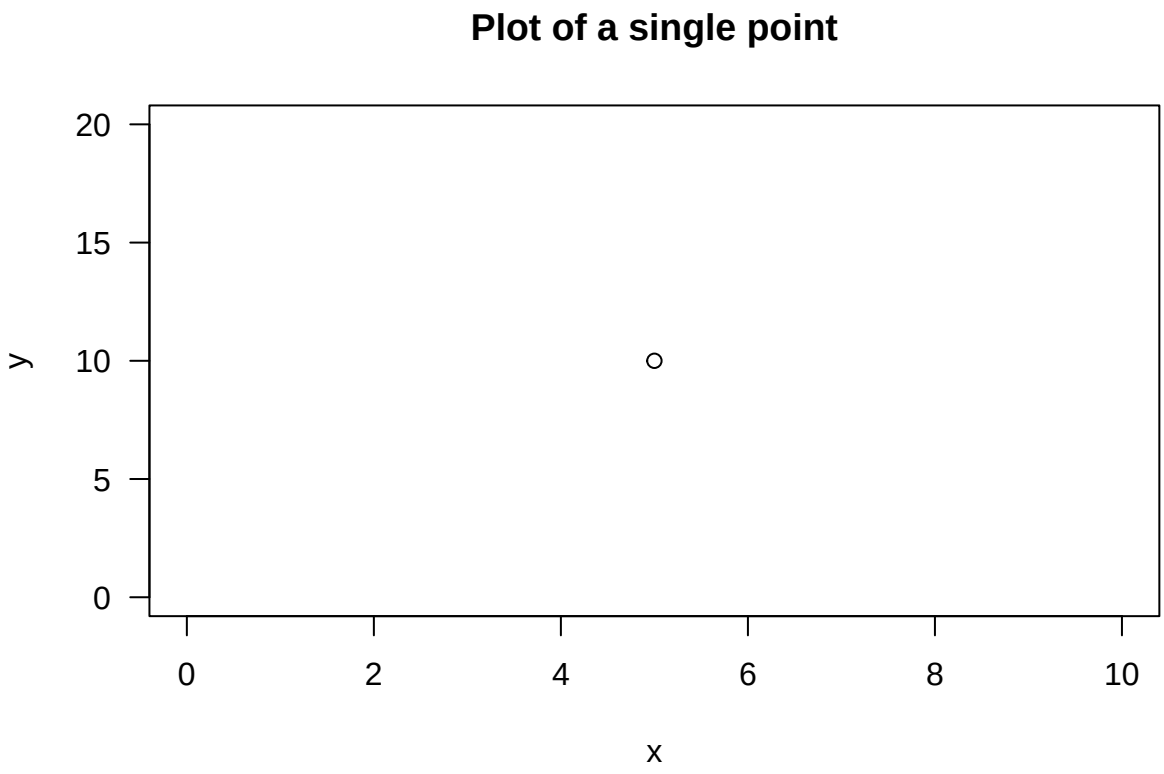


Make sure that you understand exactly how this graph differs from the original graph, and be able to explain how this was achieved.

Putting it all together

Let's put all these various techniques together:

```
plot(
    x = 5,
    y = 10,
    xlim = c(0, 10),
    ylim = c(0, 20),
    main = "Plot of a single point",
    xlab = "x",
    ylab = "y",
    las = 1
)
```



So that's how to modify aspects of the basic plot by using options.

Now let's investigate another method for modifying the display, called "graphical parameters".

Exercise 2.1: Modifying a basic plot

Plot a single point at the location (60, 240). Set the x-axis to range between 0 and 100, and the y-axis to range between 0 and 400. Be sure to explicitly specify a main title, as well as titles for the x- and y axes.

Solution

```
# Type your answer here
```

Section 3: Graphical Parameters

Main Idea *We can modify aspects of the display by using graphical parameters*

In this section, we'll see how to modify aspects of the basic plot by using graphical parameters.

Another way that we can modify the graphical display is by using *graphical parameters*.

We can set graphical parameters using the `par()` function.

There are many graphical parameters, but since this is an introductory course we won't have time to conduct a complete survey.

Instead, I'll show you two examples of graphical parameters, and if you're interested you can investigate this further in the official R documentation.

Controlling the box type

In our original graph, there is a box which surrounds the entire plotting region.

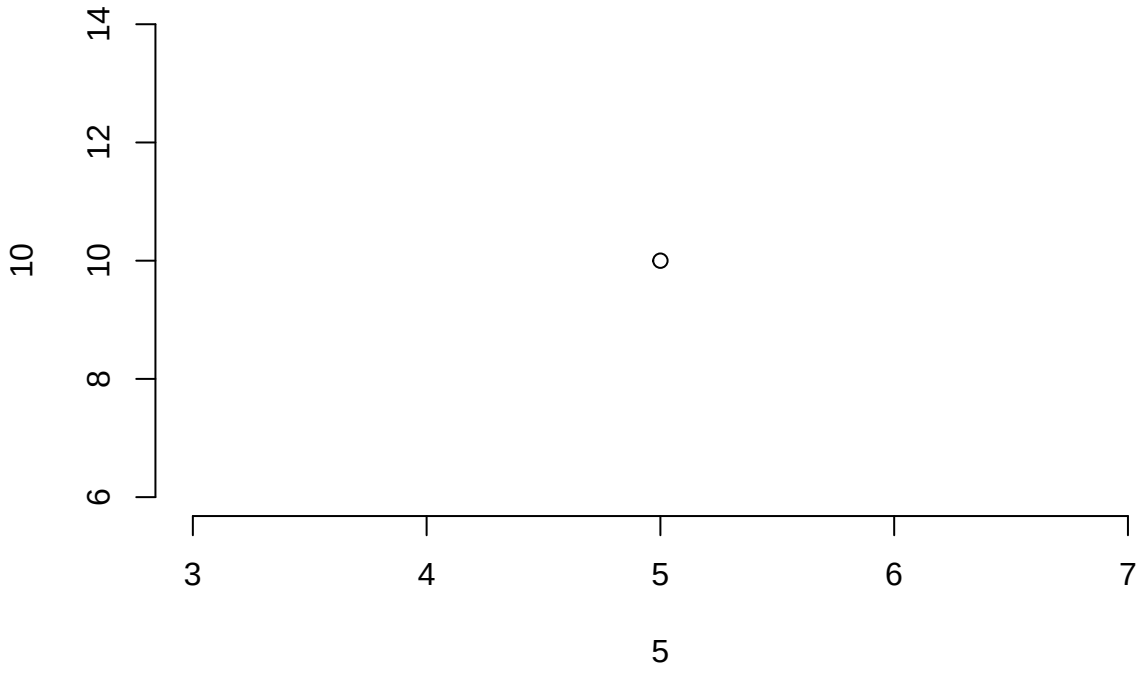
We can eliminate this box by setting the graphical parameter `bty = "n"`:

```
par( bty = "n" )  # There will be no box around the graph
```

However, by itself this code won't do anything, because we haven't tried to create a graph using the `plot()` function.

Instead, we have to call the `par()` function inside a code chunk that then draws a graph:

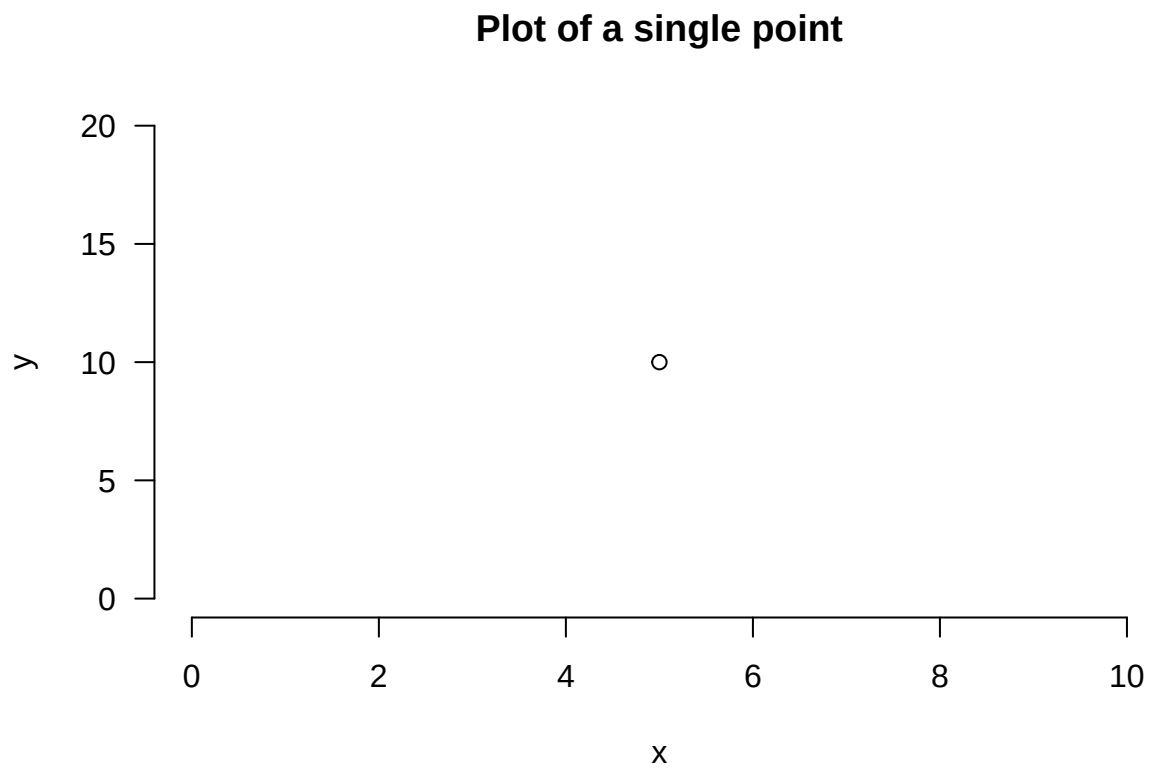
```
# First, let's set the graphical parameter:

par( bty = "n" )  # There will be no box around the graph

# Now we'll draw a simple graph:

plot(
    x = 5,
    y = 10
)
```



Make sure that you understand exactly how this graph differs from the original graph, and be able to explain how this was achieved.

Now let's try this with our more complete version:

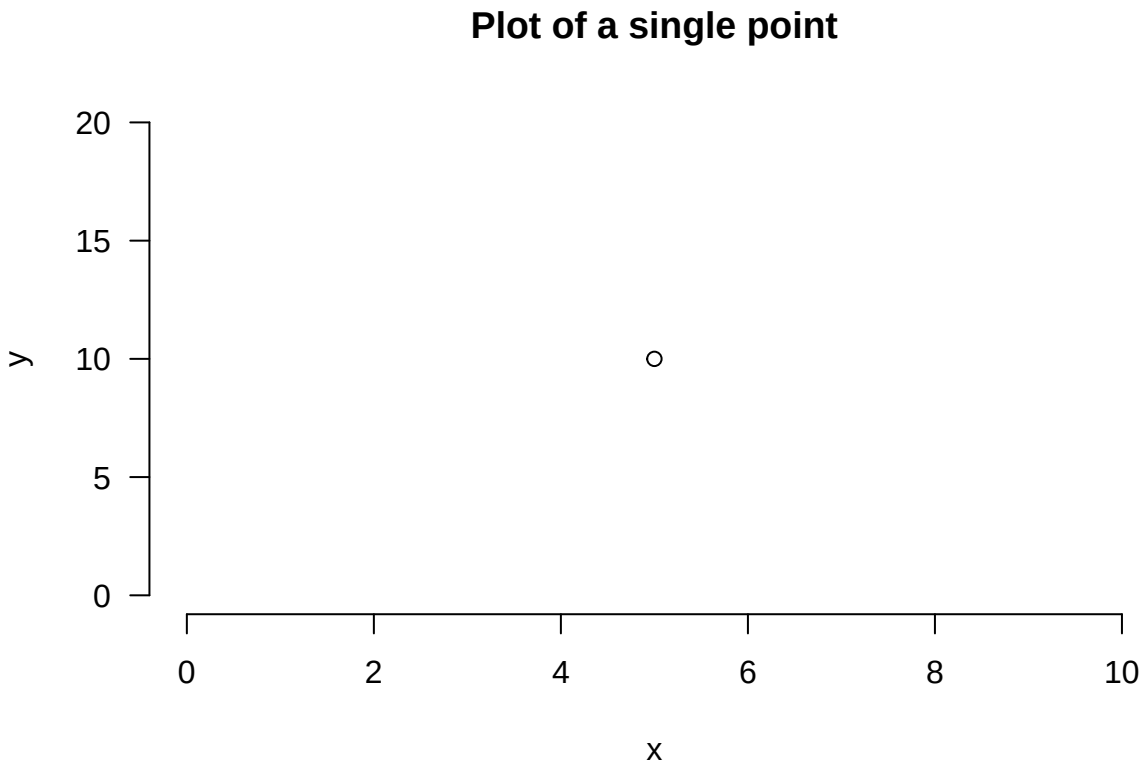
```
# First, let's set the graphical parameter:

par( bty = "n" )

# Now we can make our graph:

plot(
    x = 5,
    y = 10,
    xlim = c(0, 10),
    ylim = c(0, 20),
    main = "Plot of a single point",
    xlab = "x",
    ylab = "y",
    las = 1
)
```



In fact, in this example you don't even have to call the `par()` function, and instead can simply set the value of the `bty` graphical parameter directly from inside the `plot()` function:

```
plot(
    x = 5,
    y = 10,
    xlim = c(0, 10),
    ylim = c(0, 20),
    main = "Plot of a single point",
```

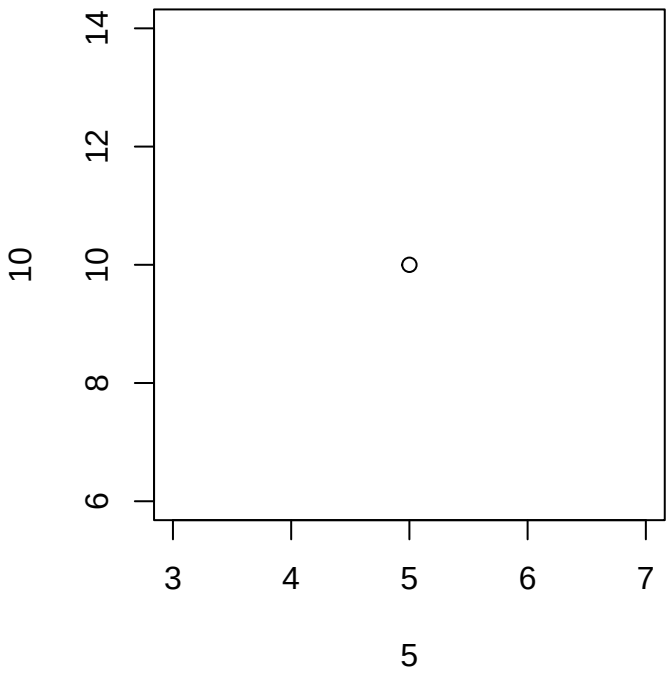
```
    xlab = "x",
    ylab = "y",
    las = 1,
    bty = "n"    ## Suppressing the box
)
```



Controlling the plot shape

We can obtain a square plot by setting the graphical parameter `pty` to the value "s", which stands for "square":

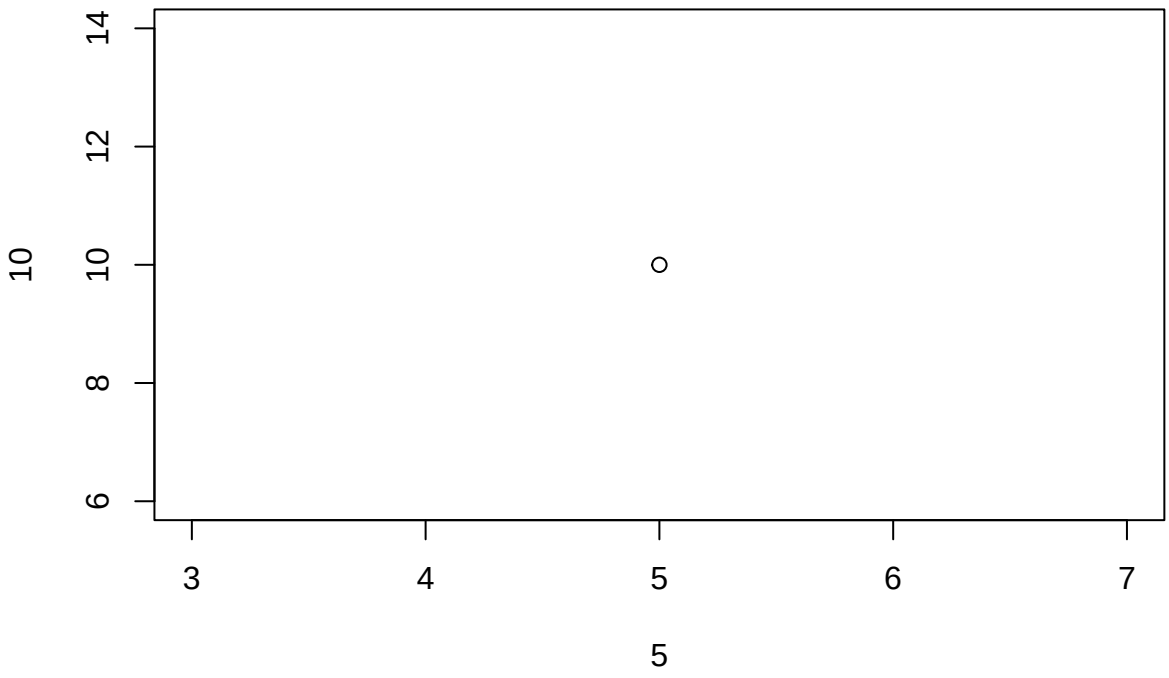
```
# First, let's set the graphical parameter pty:

par( pty = "s" )

# Now we can make our graph:

plot(
    x = 5,
    y = 10
)
```



Notice that even though the display is square, the x- and y-axes have different scales, because the ranges of the axes are different.

By the way, this is an example of where we have to use the `par()` function, and we can't set the display to a square by specifying the `pty` option from within the `plot()` function:

```
plot(
    x = 5,
    y = 10,
    pty = "s"
)
```



So that's how to modify aspects of the basic plot by using graphical parameters.

Now let's learn how to create empty displays.

Exercise 2.2: Using graphical parameters

Plot a single point at the location (60, 90).

Set both the x-axis and the y-axis to range between 0 and 100, and use a square graph shape.

Set a graphical parameter to remove the box around the display.

Be sure to include all the fancy stuff!

Solution

```
# Type your answer here
```

Section 4: Empty Displays

> **Main Idea** *We can create empty plotting displays*

In this section, we'll learn how to create empty displays.

Before we can start to create graphics, we first have to have something for us to draw on, just as a painter needs to have a surface such as a canvas or a wall in order to create a painting.

And just like a painter who starts with a blank canvas, it's often desirable to be able to create an empty plotting region.

That might seem a little strange: why would we want to create a graph that has nothing in it?

The answer is that we'll use this empty graph as a foundation to build sophisticated custom displays.

In this section, we'll examine two ways to make such a "blank canvas":

- First, we can make an empty plot with no data, but that still contains all the other components such as titles and axes.
- Second, we can create a completely blank plot, with no visible content at all.

Creating an empty plot with no data

Let's consider the first approach, where we create a plot without any actual data, but which has other components such as titles and axes.

In R, the basic function for creating a plotting region is `plot()`, but this function requires us to specify some sort of data to be plotted.

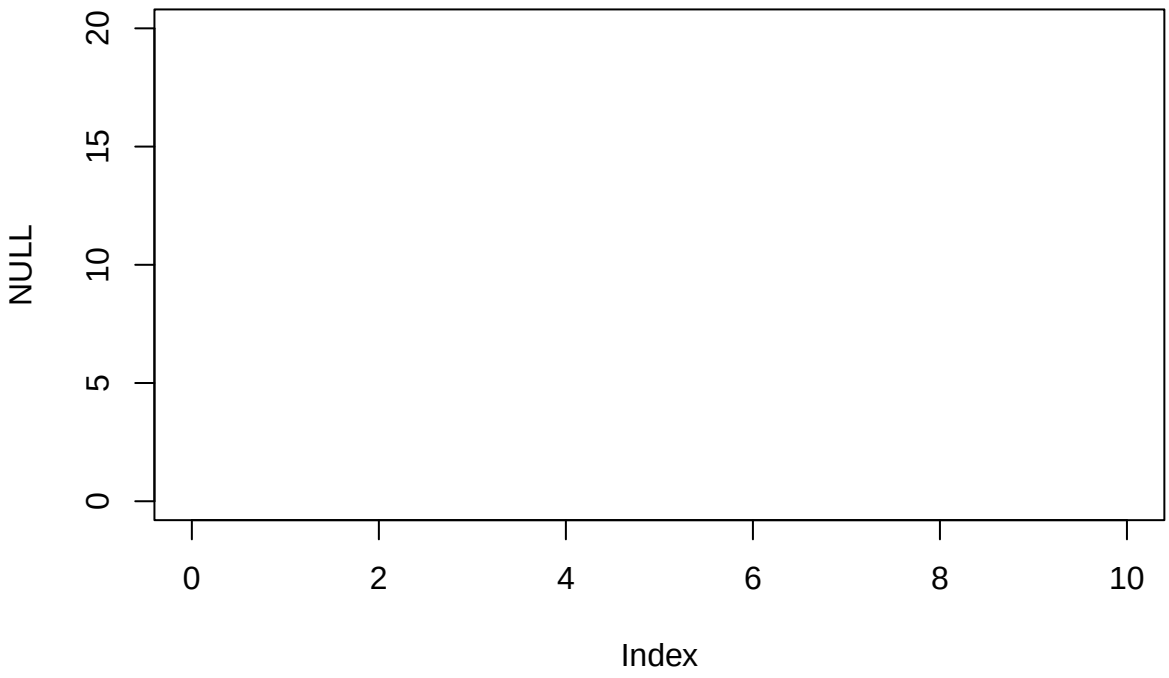
How can we use the `plot()` function to create an empty plot with no data?

The answer is that we assign the value `NULL` to the data by using the expression `x = NULL`, and then specify all the other components of the graph using standard techniques.

Let's use this technique to create an empty plot with no data.

We'll set the range of x values to go from 0 to 10, and the range of y values to go from 0 to 20:

```
plot(
    x = NULL,
    xlim = c(0, 10),
    ylim = c(0, 20)
)
```



It's important to remember that if you want to create an empty plot with no data, then you **must** specify the ranges of the x- and y-axes.

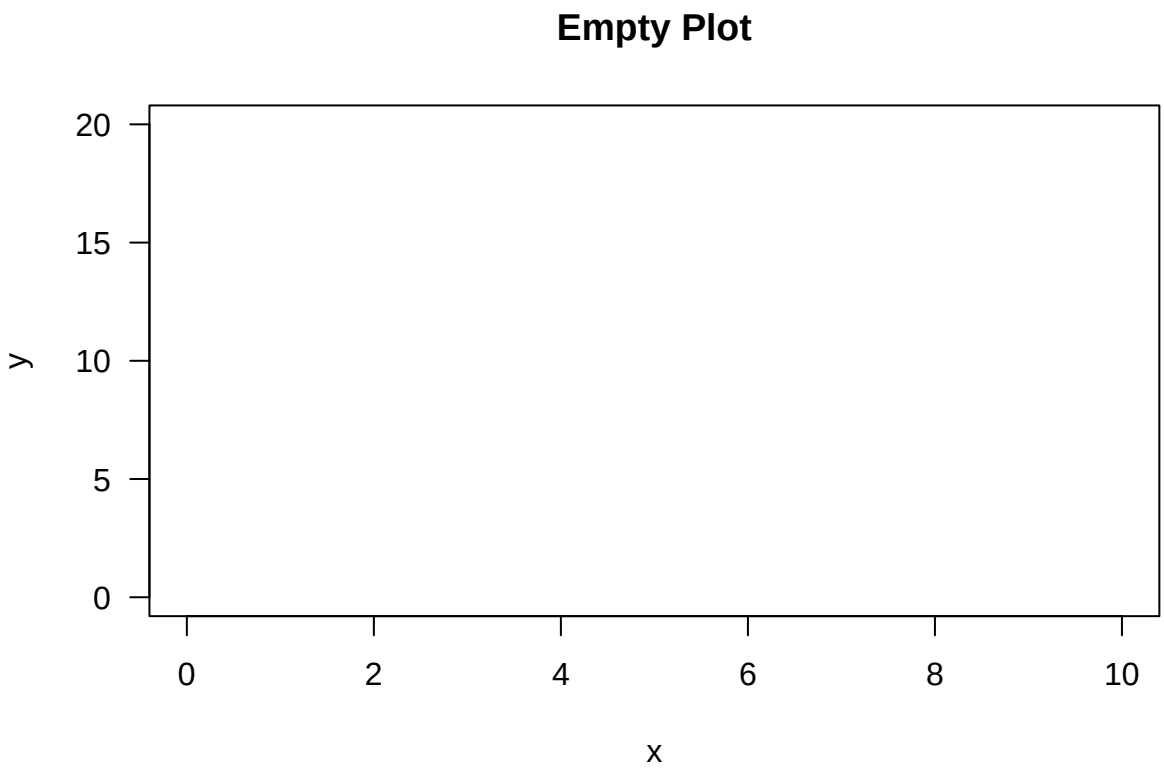
If you think about it, this is very sensible: when you create a plot with data, then R can examine the values in your data and make sensible guesses for the ranges of the axes.

But if you create the plot without data, then R has no way to determine what the ranges of the axes should be.

Thus, if you want to create an empty plot with no data, then you have to explicitly specify the ranges of the axes.

Now let's put in all the extras:

```
plot(
    x = NULL,
    xlim = c(0, 10),
    ylim = c(0, 20),
    main = "Empty Plot",
    xlab = "x",
    ylab = "y",
    las = 1
)
```



Notice that there's still a lot of content in this supposedly "empty" graph: it contains a main title, axes, axis titles, and the surrounding box.

Most importantly, the ranges of the x- and y-axes have been determined.

In our course I'm going to call this sort of plot an "empty graph with no data", and that means that we can still specify all the other components.

This terminology is not standard, but there doesn't seem to be any commonly accepted jargon for this concept, so I made the term up so that we can have a simple and consistent way to refer to this technique.

Exercise 2.3: Creating an empty plot with no data

Create an empty graph with no data where x ranges from 0 to 20 and y ranges from 0 to 5.

Solution

```
# Type your answer here
```

Creating a completely blank display

As we've seen, when we create an empty plot with no data, there is still a lot of visual content on display.

What can we do if we want to eliminate all of this remaining content, and just create a completely blank display?

First, we can eliminate both the box and the axes as well by using the option `axes = FALSE`:

```
# We can draw the graph without axes or an enclosing box:

plot(
    x = NULL,
    xlim = c(0, 10),
    ylim = c(0, 20),
    axes = FALSE
)
```

Finally, we can eliminate the axis titles by assigning an empty string to the `xlab =` and `ylab =` parameters:

```
plot(
    x = NULL,
    xlim = c(0, 10),
    ylim = c(0, 20),
    xlab = "",
    ylab = "",
    axes = FALSE
)
```

I will call a plotting region such as this a "completely blank display".

Notice that although we have eliminated all visual content, this plotting region does indeed have structure, because we've specified the ranges of the x- and y-axes.

So, even though it seems as though we've gone through a lot of work to make nothing, we actually have created something that will later be useful.

Remember, when I say "create an empty plot with no data", that means something very specific, and when I say "create a completely blank display" that means something else.

So that's how to create empty displays.

Now let's review what we've learned in this module.

Exercise 2.4: Creating a completely blank display

Create a completely blank display where x ranges from 0 to 20 and y ranges from 0 to 5.

Solution

Module Review

In this module, we learned about the basic features of graphics plots.

- In Section 1, we created a basic plot with a single point.

- In Section 2, we learned how to modify aspects of the basic plot by using options.
- In Section 3, we saw how to modify aspects of the basic plot by using graphical parameters.
- In section 4, we learned how to create empty displays.

Now that you've completed this module, you should be able to:

- Create a basic plot with a single point.
- Control the range of the x- and y-axes.
- Specify the text of the main title.
- Modify the text of the x- and y-axis titles.
- Suppress the display of the outside box.
- Specify a square display.
- Create an empty plot with no data.
- Create a completely blank display.

In this module, we learned about 2 new built-in R functions:

- `plot()`
- `par()`

All right, that's it for Module 2: Graphics Plots.

Now let's move on to Module 3: Points.

Solutions to the Exercises

Exercise 2.1: Modifying a basic plot

Plot a single point at the location (60, 240).

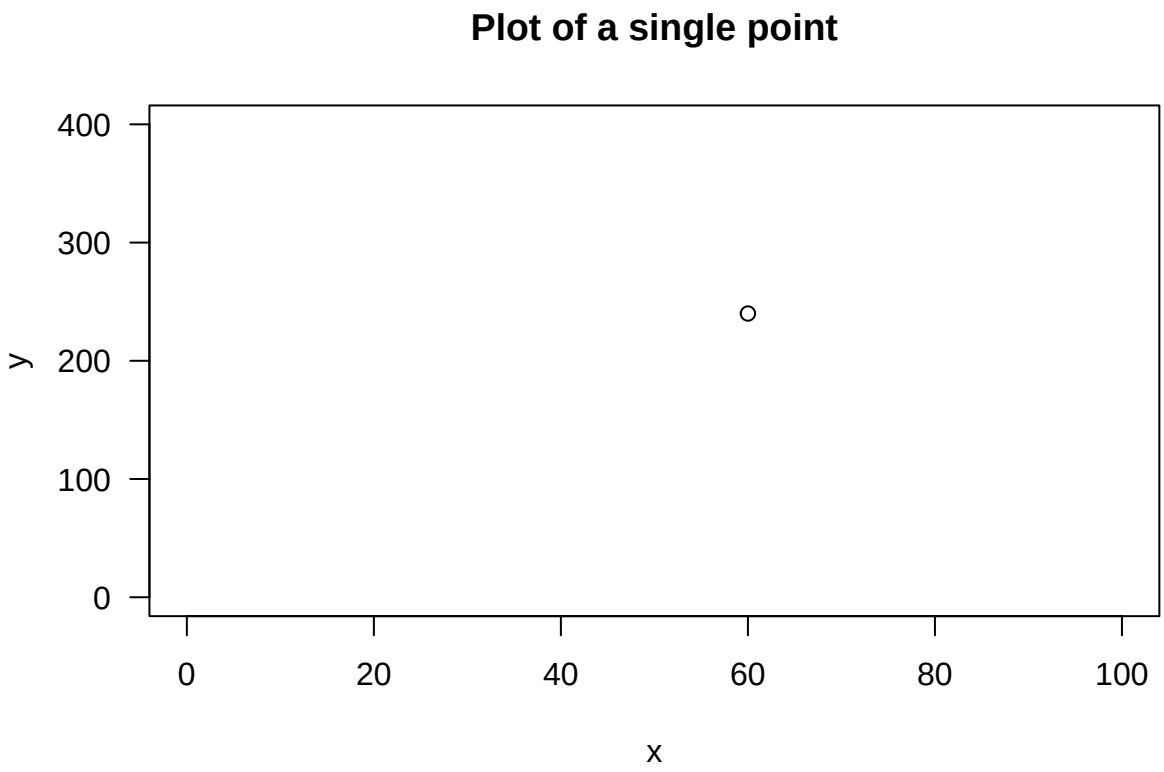
Set the x-axis to range between 0 and 100, and the y-axis to range between 0 and 400.

Be sure to put in all the extra stuff!

Solution

Here's my solution:

```
plot(
    x = 60,
    y = 240,
    xlim = c(0, 100),
    ylim = c(0, 400),
    main = "Plot of a single point",
    xlab = "x",
    ylab = "y",
    las = 1
)
```



Exercise 2.2: Using graphical parameters

Plot a single point at the location (60, 90).

Set both the x-axis and the y-axis to range between 0 and 100, and use a square graph shape.

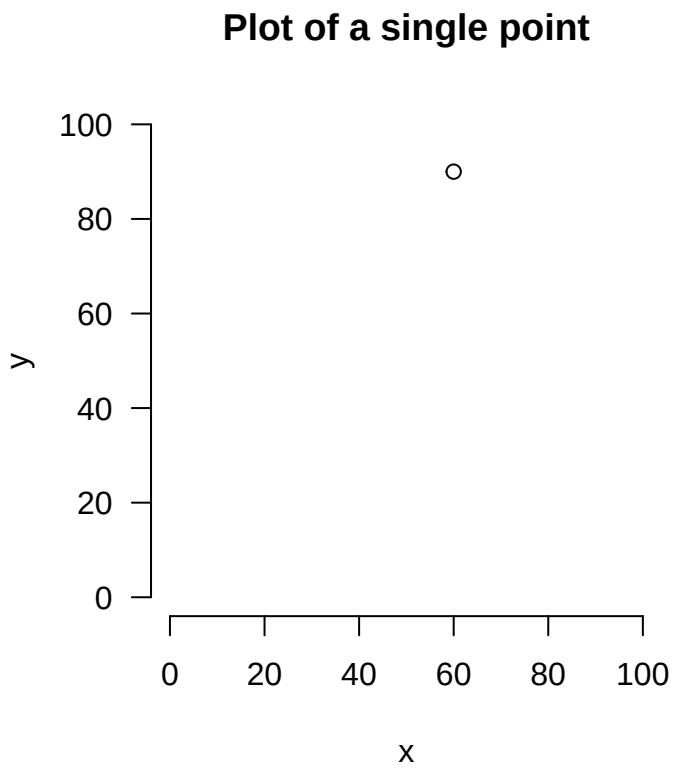
Set a graphical parameter to remove the box around the display.

Be sure to include all the fancy stuff!

Solution

Here's my solution:

```
par( bty = "n" )

par( pty = "s" )

plot(
    x = 60,
    y = 90,
    xlim = c(0, 100),
    ylim = c(0, 100),
    main = "Plot of a single point",
    xlab = "x",
    ylab = "y",
    las = 1
)
```



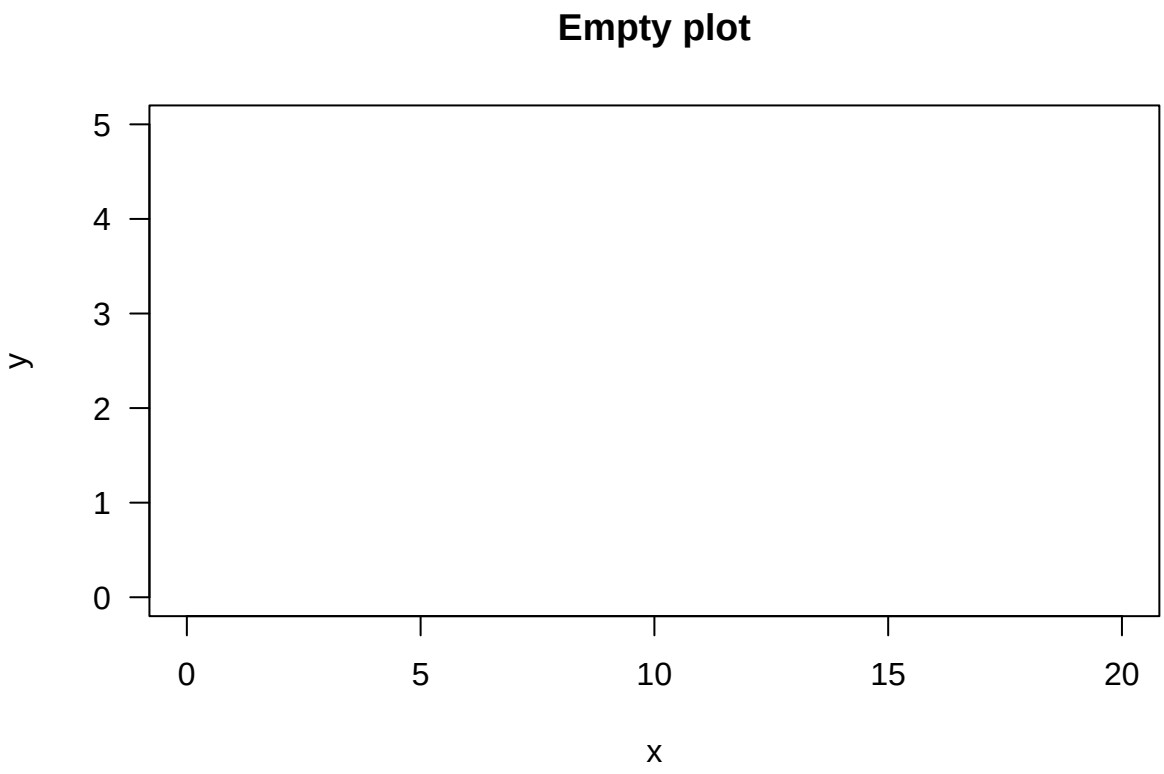
Exercise 2.3: Creating an empty plot with no data

Create an empty graph with no data where x ranges from 0 to 20 and y ranges from 0 to 5.

Solution

Here's my solution:

```
plot(
    x = NULL,
    xlim = c(0, 20),
    ylim = c(0, 5),
    main = "Empty plot",
    xlab = "x",
    ylab = "y",
    las = 1
)
```



Exercise 2.4: Creating a completely blank display

Create a completely blank plotting region where x ranges from 0 to 20 and y ranges from 0 to 5.

Solution

```
plot(
    x = NULL,
    xlim = c(0, 20),
    ylim = c(0, 5),
    xlab = "",
    ylab = "",
    axes = FALSE
)
```