DATA SOCIETY®

Static Visualization

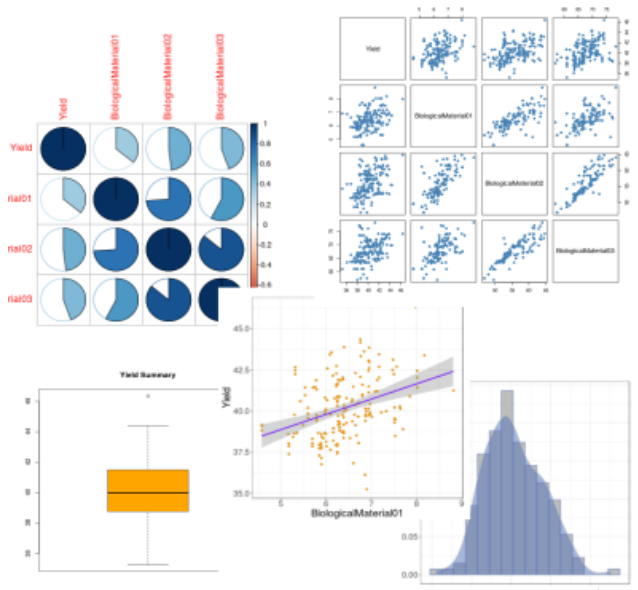
"One should look for what is and not what he thinks should be."
-Albert Einstein.

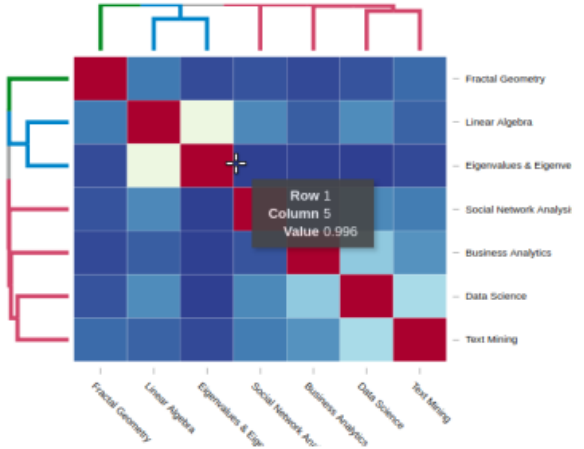
Module completion checklist

Objective	Complete
Define exploratory data analysis (EDA) and its cycle	
Define and differentiate between static and interactive visualizations	
Choose when to use univariate plots to illustrate patterns in data	
Construct univariate plots using base R	
Choose when to use bivariate plots to illustrate patterns in data	
Construct bivariate plot using base R	
Choose when to use multivariate plots to illustrate patterns in data	
Construct multivariate plot (scatterplot matrix) using base R	
Measure and observe correlation using corrplot package	
Formulate the process of using ggplot2 to build complex plots	
Build univariate plots using various geom layers with ggplot2	
Build bivariate plots using various geom layers with ggplot2	

Why build a visualization?

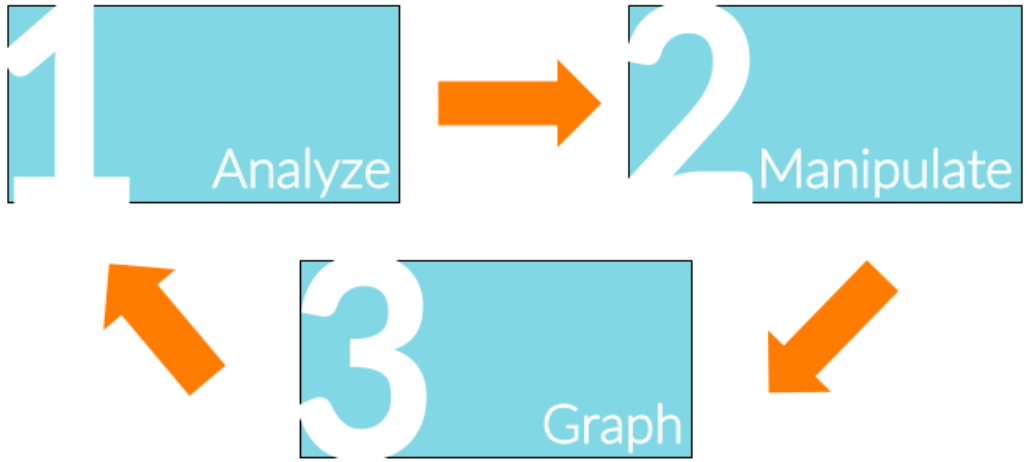
- To communicate your ideas?
- To better understand your data?
- To discover a new insight?
- Visualization is a great way to do exploratory data analysis (**EDA**)





Exploratory data analysis

- R is a powerful tool for EDA because the graphics tie in with the functions used to analyze data. You can create graphs without breaking your train of thought as you explore your data. Visualization is an iterative process and consists of a few steps:
  - Analyze
  - Manipulate
  - Graph
  - Repeat



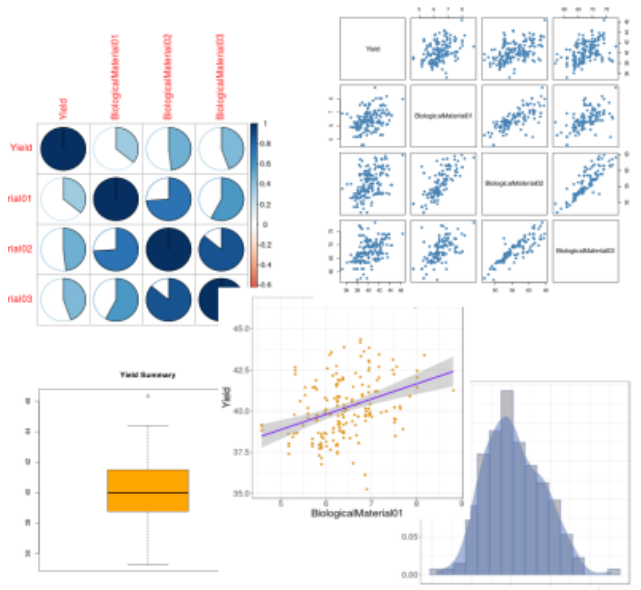
Module completion checklist

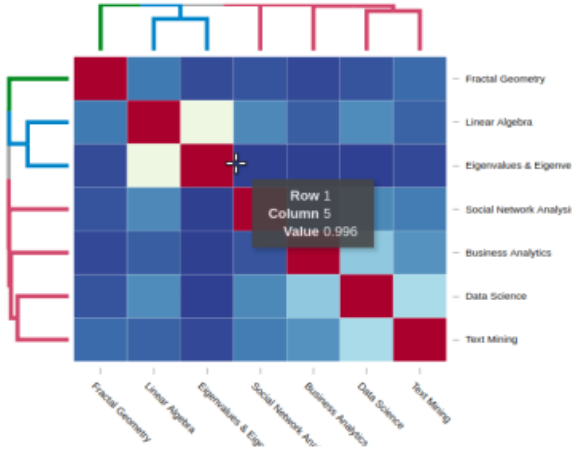
Objective	Complete
Define exploratory data analysis (EDA) and its cycle	✔
Define and differentiate between static and interactive visualizations	
Choose when to use univariate plots to illustrate patterns in data	
Construct univariate plots using base R	
Choose when to use bivariate plots to illustrate patterns in data	
Construct bivariate plot using base R	
Choose when to use multivariate plots to illustrate patterns in data	
Construct multivariate plot (scatterplot matrix) using base R	
Measure and observe correlation using corrplot package	
Formulate the process of using ggplot2 to build complex plots	
Build univariate plots using various geom layers with ggplot2	
Build bivariate plots using various geom layers with ggplot2	

What can you ultimately do in R?

- Create multiple visualization types like:
  - Basic plots & composite graphs
  - Maps
  - Dynamic visualizations
  - Interactive charts & dashboards
  - 3D graphics
- Make reproducible visualizations
- Save and share your work in a variety of file formats and with adjustable image quality (`SVG`, `PNG`, `JPEG`, `PDF`, etc.)

Use `??visualization` to see a list of help pages, vignettes, and code demos

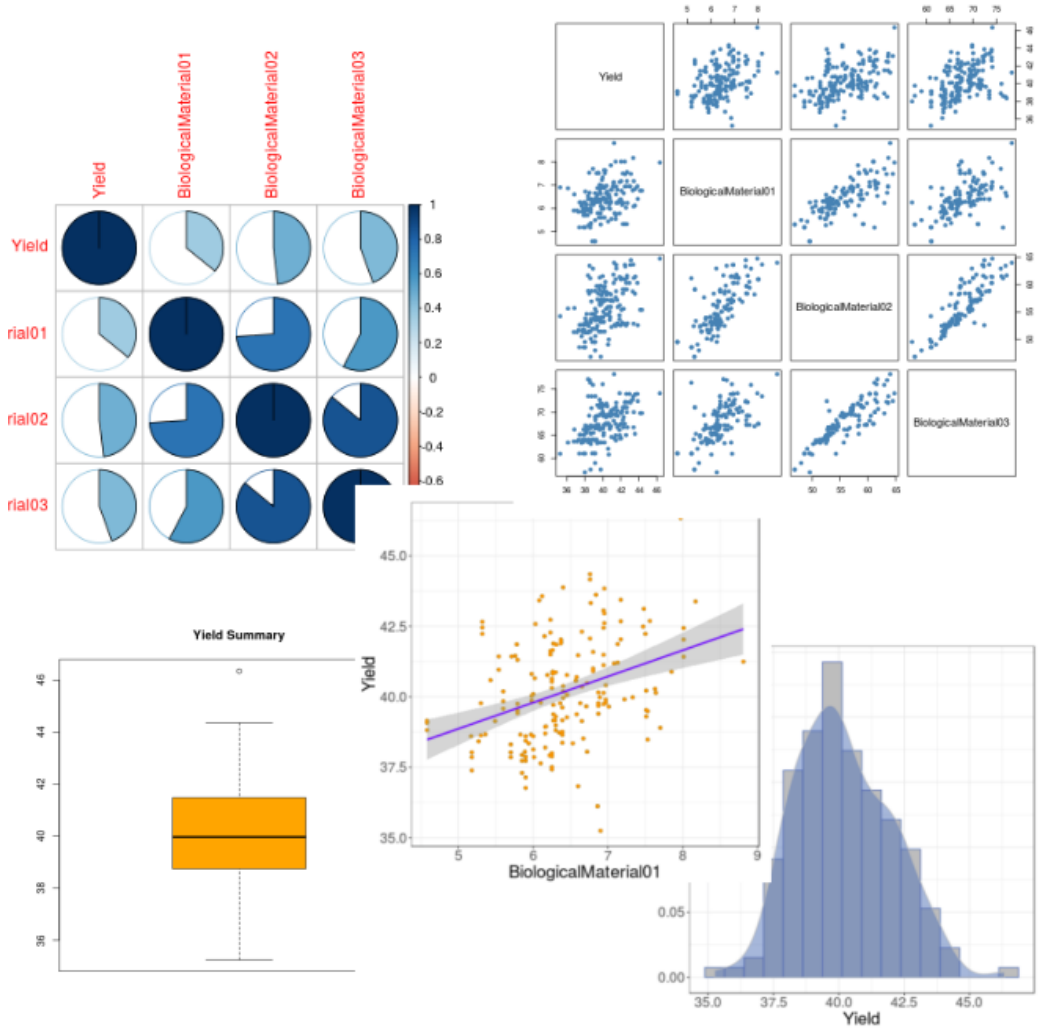




Visualizing data in R

Static plots

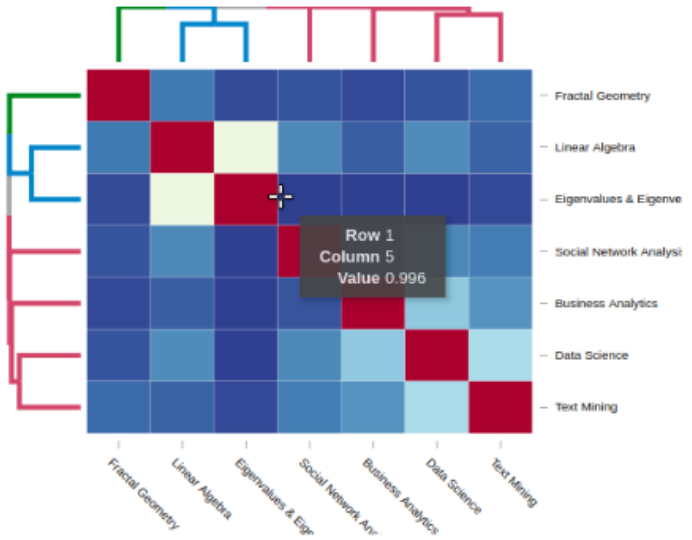
1. **Display** data **only**, **user cannot interact** with visualization
2. Available through base R and multitudes of packages (e.g. `ggplot2`, `corrplot`)
3. High quality (all R visualizations are made in scalable vector graphics format, or `SVG`), can be saved as `SVG`, `PNG`, `JPEG`, `BMP`, `PDF`
4. The best way to display patterns in data for **printed publications**

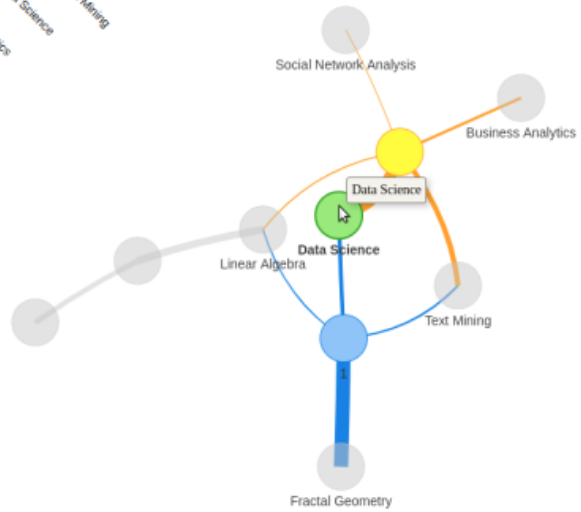


Visualizing data in R

Interactive visualizations

1. **Display** data **and user can interact** with visualization by clicking, hovering, dragging, zooming, etc.
2. Available through many packages (e.g. `highcharter`, `plotly`, `htmlwidgets`)
3. High quality (all graphical elements are in SVG format), rendered as `HTML` pages, can be saved as `HTML` documents and opened through a browser
4. The best way to display patterns in data for **web publications**, on **websites**, in **web applications**





Module completion checklist

Objective	Complete
Define exploratory data analysis (EDA) and its cycle	✔
Define and differentiate between static and interactive visualizations	✔
Choose when to use univariate plots to illustrate patterns in data	
Construct univariate plots using base R	
Choose when to use bivariate plots to illustrate patterns in data	
Construct bivariate plot using base R	
Choose when to use multivariate plots to illustrate patterns in data	
Construct multivariate plot (scatterplot matrix) using base R	
Measure and observe correlation using corrplot package	
Formulate the process of using ggplot2 to build complex plots	
Build univariate plots using various geom layers with ggplot2	
Build bivariate plots using various geom layers with ggplot2	

Directory settings

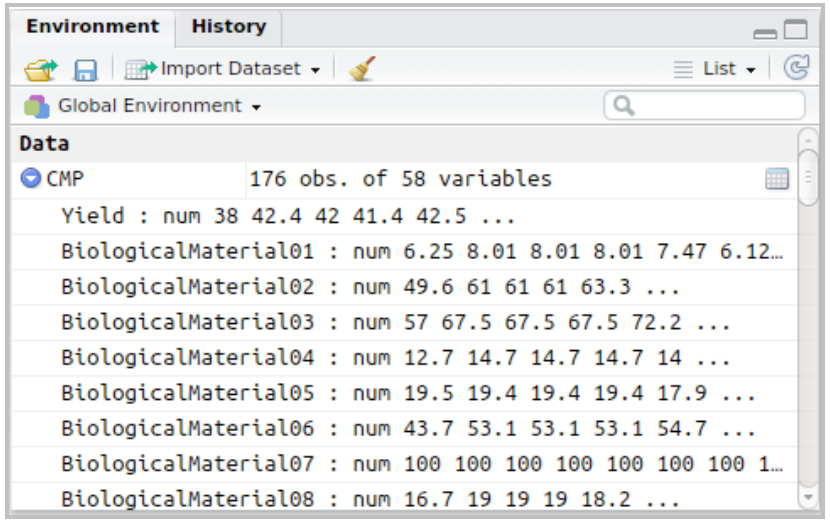
- In order to maximize the efficiency of your workflow, you may want to encode your directory structure into `variables`
- Let the `main_dir` be the variable corresponding to your `af-werx` folder

```
# Set `main_dir` to the location of your `af-werx` folder (for Mac/Linux).
main_dir = "~/Desktop/af-werx"

# Set `main_dir` to the location of your `af-werx` folder (for Windows).
main_dir = "C:/Users/[username]/Desktop/af-werx"

# Make `data_dir` from the `main_dir` and
# remainder of the path to data directory.
data_dir = paste0(main_dir, "/data")

# Make `plots_dir` from the `main_dir` and
# remainder of the path to plots directory.
plot_dir = paste0(main_dir, "/plots")
```

Loading dataset for EDA

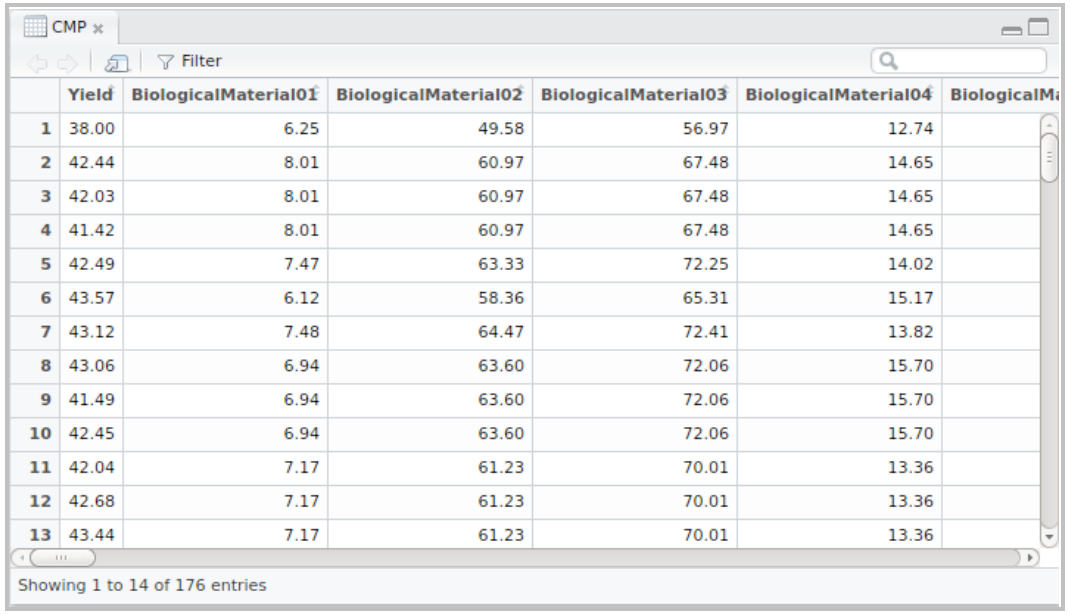
- Let's load the dataset from our `data_dir` into R's environment

```
# Set working directory to where we store data.
setwd(data_dir)

# Read CSV file called
"ChemicalManufacturingProcess.csv"
CMP =
read.csv("ChemicalManufacturingProcess.csv",
                header = TRUE,
                stringsAsFactors = FALSE)
```



- The dataset consists of `176` observations and `58` variables

```
# View CMP dataset in the tabular data explorer.
View(CMP)
```



	Yield	BiologicalMaterial01	BiologicalMaterial02	BiologicalMaterial03	BiologicalMaterial04	BiologicalMa
1	38.00	6.25	49.58	56.97	12.74	
2	42.44	8.01	60.97	67.48	14.65	
3	42.03	8.01	60.97	67.48	14.65	
4	41.42	8.01	60.97	67.48	14.65	
5	42.49	7.47	63.33	72.25	14.02	
6	43.57	6.12	58.36	65.31	15.17	
7	43.12	7.48	64.47	72.41	13.82	
8	43.06	6.94	63.60	72.06	15.70	
9	41.49	6.94	63.60	72.06	15.70	
10	42.45	6.94	63.60	72.06	15.70	
11	42.04	7.17	61.23	70.01	13.36	
12	42.68	7.17	61.23	70.01	13.36	
13	43.44	7.17	61.23	70.01	13.36	

Subsetting data

- In this module, we will explore a subset of this dataset, which includes the following variables
  - **yield**
  - **3 material** variables, and
  - **3 process** variables

CMP ×

Filter

	Yield	BiologicalMaterial01	BiologicalMaterial02	BiologicalMaterial03
1	38.00	6.25	49.58	56.97
2	42.44	8.01	60.97	67.48
3	42.03	8.01	60.97	67.48
4	41.42	8.01	60.97	67.48
5	42.49	7.47	63.33	72.25
6	43.57	6.12	58.36	65.31
7	43.12	7.48	64.47	72.41
8	43.06	6.94	63.60	72.06
9	41.49	6.94	63.60	72.06

Showing 1 to 10 of 176 entries

...

CMP ×

Filter

ManufacturingProcess01	ManufacturingProcess02	ManufacturingProcess03
NA	NA	NA
0.0	0.0	NA
0.0	0.0	NA
0.0	0.0	NA
10.7	0.0	NA
12.0	0.0	NA
11.5	0.0	1.56
12.0	0.0	1.55
12.0	0.0	1.56

Subsetting data

```
# Let's make a vector of column indices we would like to save.
column_ids = c(1:4,  #<- concatenate a range of IDs
               14:16)#<- with another a range of IDs
column_ids
```

```
[1]  1  2  3  4 14 15 16
```

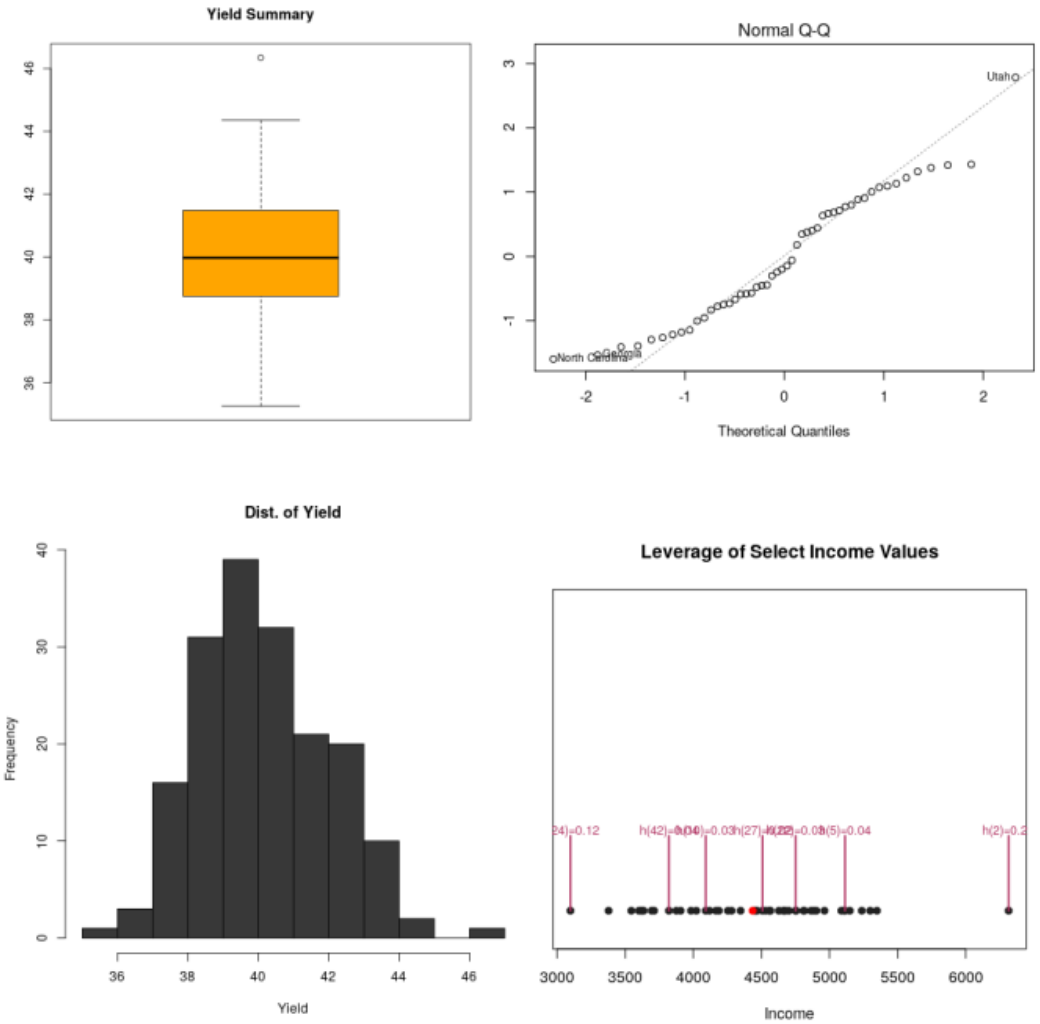
```
# Let's save the subset into a new variable.
CMP_subset = CMP[ , column_ids]
str(CMP_subset)
```

```
'data.frame':   176 obs. of  7 variables:
 $ Yield                 : num  38 42.4 42 41.4 42.5 ...
 $ BiologicalMaterial01  : num  6.25 8.01 8.01 8.01 7.47 6.12 7.48 6.94 6.94 6.94 ...
 $ BiologicalMaterial02  : num  49.6 61 61 61 63.3 ...
 $ BiologicalMaterial03  : num  57 67.5 67.5 67.5 72.2 ...
 $ ManufacturingProcess01: num  NA 0 0 0 10.7 12 11.5 12 12 12 ...
 $ ManufacturingProcess02: num  NA 0 0 0 0 0 0 0 0 0 ...
 $ ManufacturingProcess03: num  NA NA NA NA NA NA 1.56 1.55 1.56 1.55 ...
```

Univariate plots

Use cases

- Univariate plots are used to visualize data distribution in a **single variable**
- They are used primarily in the initial stages of EDA when we would like **to learn more about individual variables** in our data
- They are also used **in combination with other univariate plots to compare data distributions of different variables**
- Univariate plots include the following popular graphs: `boxplot`, `histogram`, `density curve`, `dot plot`, `QQ plot`, `bar plot`

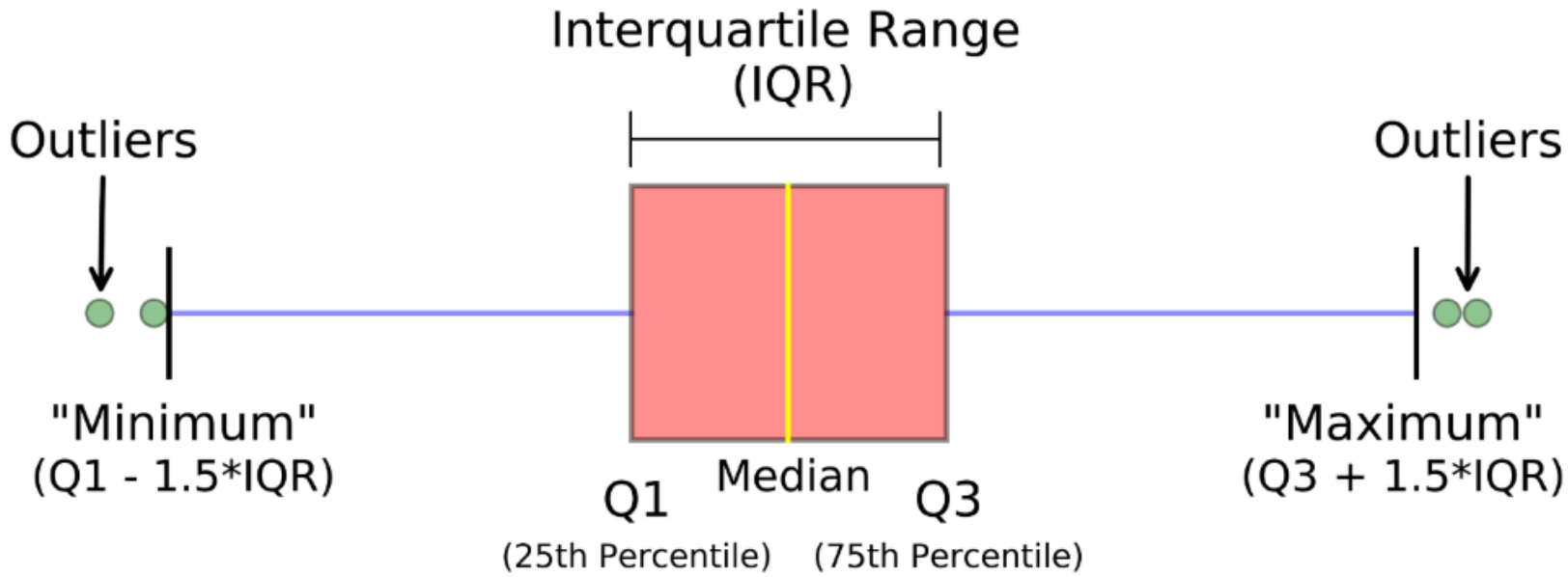


Module completion checklist

Objective	Complete
Define exploratory data analysis (EDA) and its cycle	✔
Define and differentiate between static and interactive visualizations	✔
Choose when to use univariate plots to illustrate patterns in data	✔
Construct univariate plots using base R	
Choose when to use bivariate plots to illustrate patterns in data	
Construct bivariate plot using base R	
Choose when to use multivariate plots to illustrate patterns in data	
Construct multivariate plot (scatterplot matrix) using base R	
Measure and observe correlation using corrplot package	
Formulate the process of using ggplot2 to build complex plots	
Build univariate plots using various geom layers with ggplot2	
Build bivariate plots using various geom layers with ggplot2	

Univariate plots: boxplot

- Boxplot shows the distribution of the variable



Univariate plots: boxplot

Numerical summary

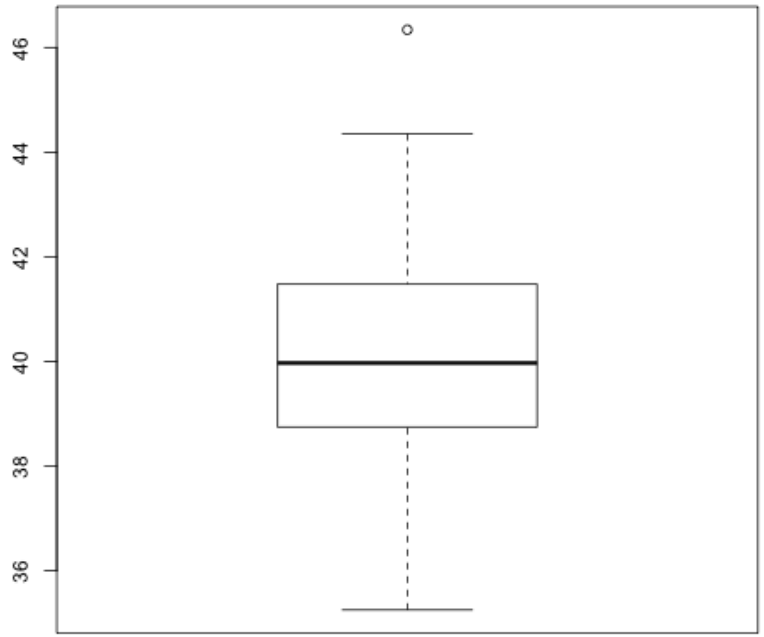
```
# Yield summary.
summary(CMP_subset$Yield)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  35.25   38.75   39.97   40.18   41.48   46.34
```

- Min value (`35.25`): bottom `whisker` of the boxplot
- Max value (`46.34`): circle (it's an outlier!)
- 1st quartile value (`38.75`): bottom of the box
- Median(2nd quartile) value (`39.97`): line in the box
- 3rd quartile value (`41.48`): top of the box

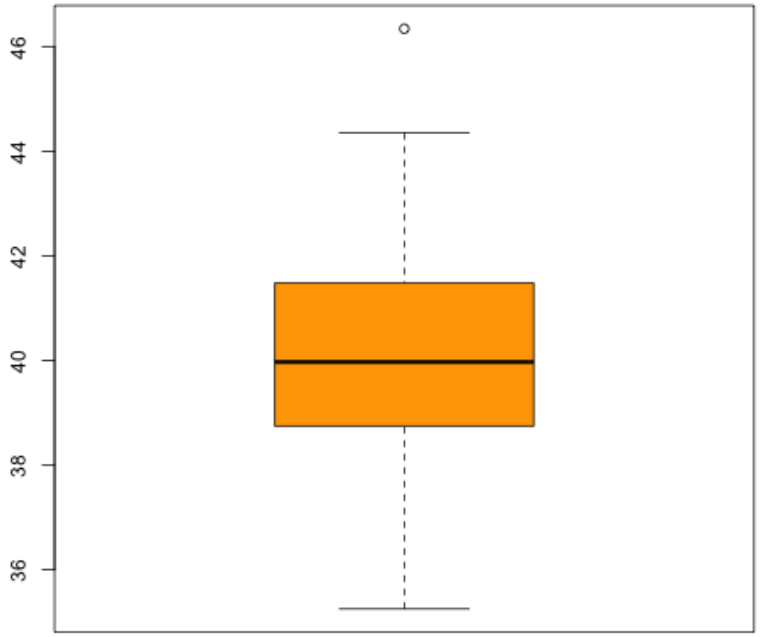
Graphical summary

```
# Univariate plot: box-and-whisker plot.
boxplot(CMP_subset$Yield)
```

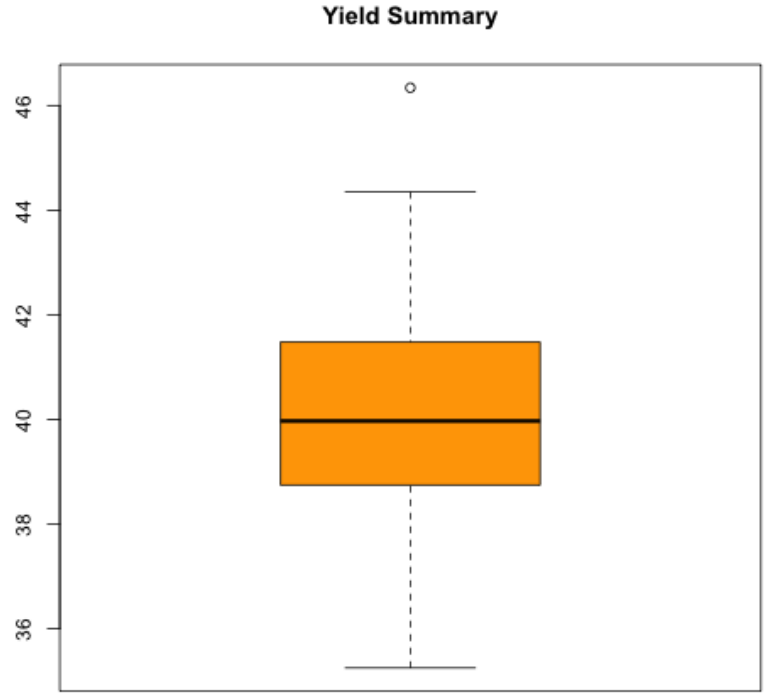


Univariate plots: boxplot

```
# Customize your boxplot.
boxplot(CMP_subset$Yield,
        col = "orange") #<- set color
```




```
# Customize your boxplot.
boxplot(CMP_subset$Yield,
        col = "orange",
        main = "Yield Summary") #<- add title
```

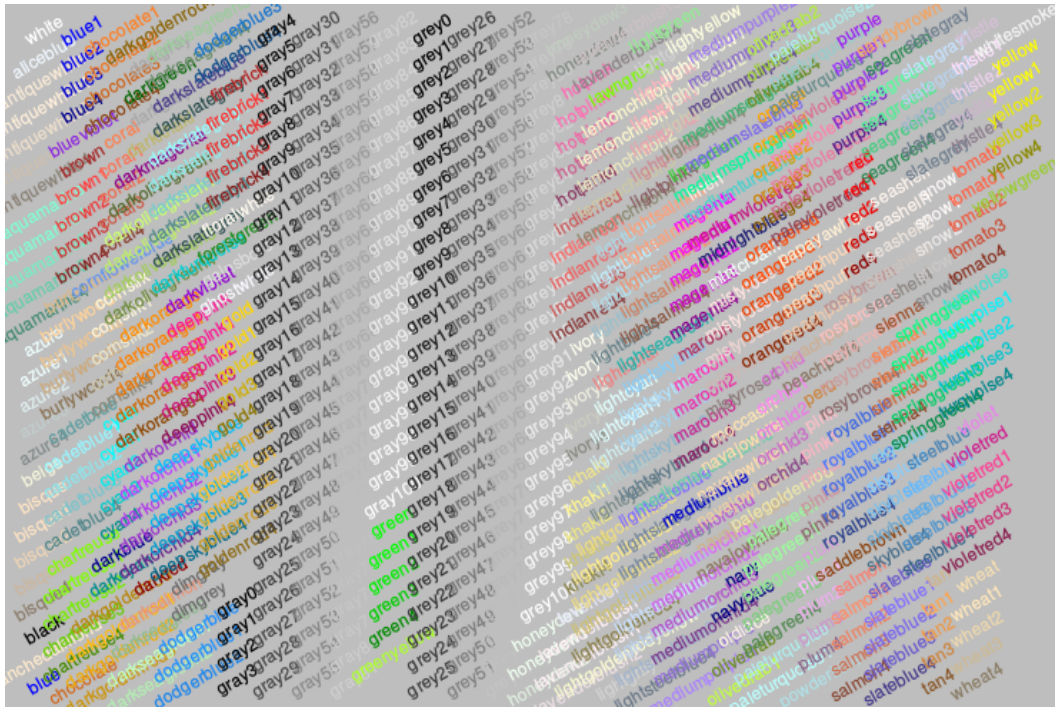


R base colors

```
# R has many colors in its native
# `grDevices` package.
?colors

# There are a total of 657 colors in
# the `colors()` vector.
str(colors())
```

```
 chr [1:657] "white" "aliceblue" "antiquewhite"
"antiquewhite1" ...
```

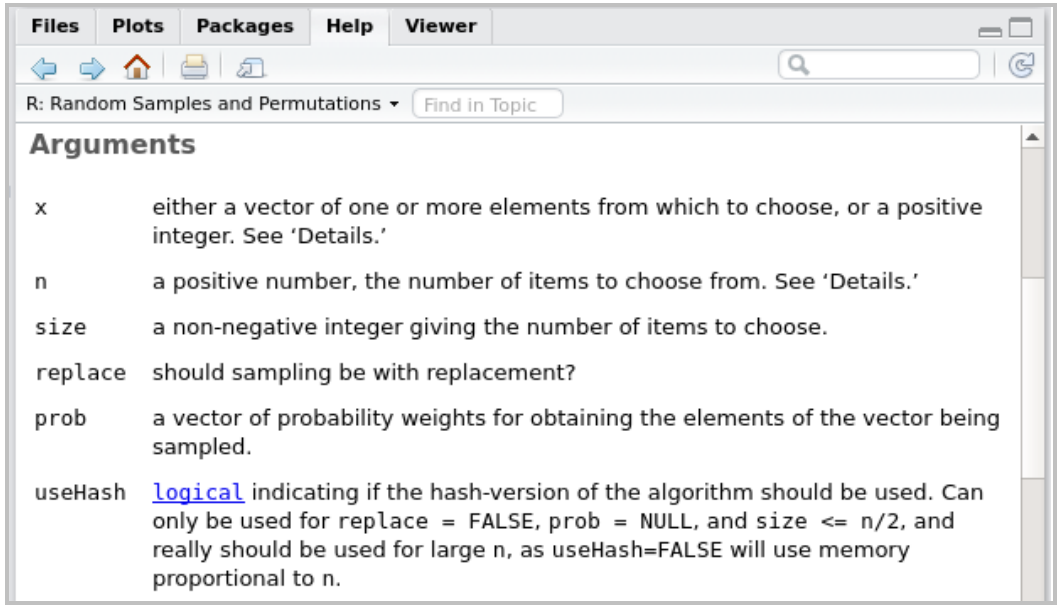
```
demo("colors")
```



Make a sample of colors for variables

```
?sample
```

1. `x` is a vector from which to sample elements
2. `n` is a number to indicate how many elements to sample



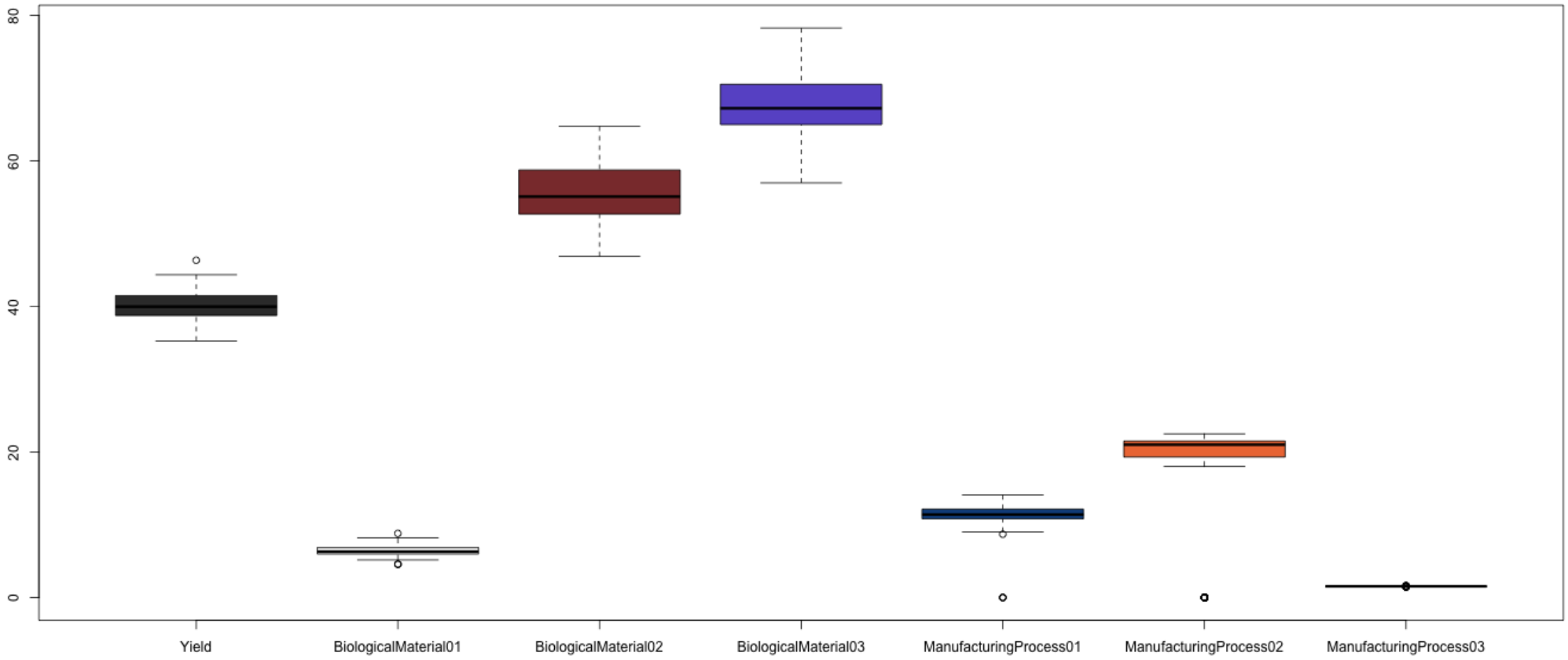
Make a sample of colors for variables

```
# Set random seed to get the same sample every time!
set.seed(1)

# We have as many variables as columns in our data.
# Save number of columns to a variable using `ncol` function.
n_cols = ncol(CMP_subset)

# Pick a sample of colors for each
# variable in our dataset.
col_sample = sample(colors(),#<- vector of colors
                    n_cols)  #<- n elements to sample
col_sample
```

```
[1] "gray22"      "gray92"      "indianred4"  "slateblue"   "dodgerblue4"
[6] "sienna2"     "steelblue"
```

Compare variables: boxplots combined

```
# Make a boxplot of all variables and give a vector of colors for each of them.
boxplot(CMP_subset, col = col_sample)
```



Univariate plots: histogram

- Histogram is a graphical representation of a numerical variable
- It is helpful in understanding the spread of the data
- We will need to analyze the distribution of the data inorder to make any assumptions or decisions about the data

Univariate plots: histogram

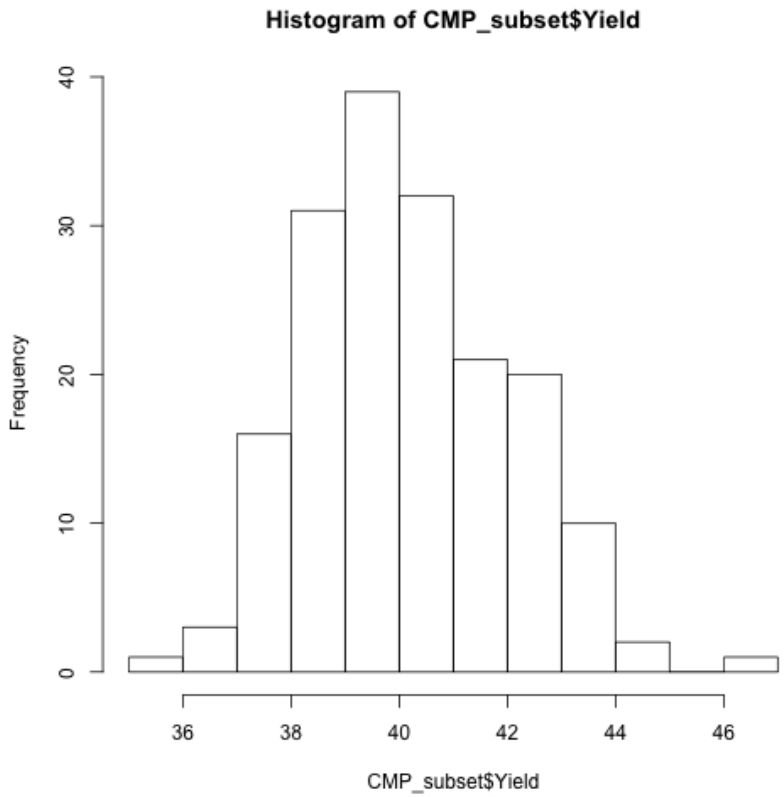
Numerical summary

```
# Histogram data without plot.
hist(CMP_subset$Yield, plot = FALSE)
```

```
$breaks
 [1] 35 36 37 38 39 40 41 42 43 44 45 46 47

$counts
 [1]  1  3 16 31 39 32 21 20 10  2  0  1

$density
 [1] 0.005681818 0.017045455 0.090909091
0.176136364 0.221590909
 [6] 0.181818182 0.119318182 0.113636364
0.056818182 0.011363636
[11] 0.000000000 0.005681818

$mids
 [1] 35.5 36.5 37.5 38.5 39.5 40.5 41.5 42.5
43.5 44.5 45.5 46.5

$xname
[1] "CMP_subset$Yield"

$equidist
[1] TRUE

attr(,"class")
[1] "histogram"
```

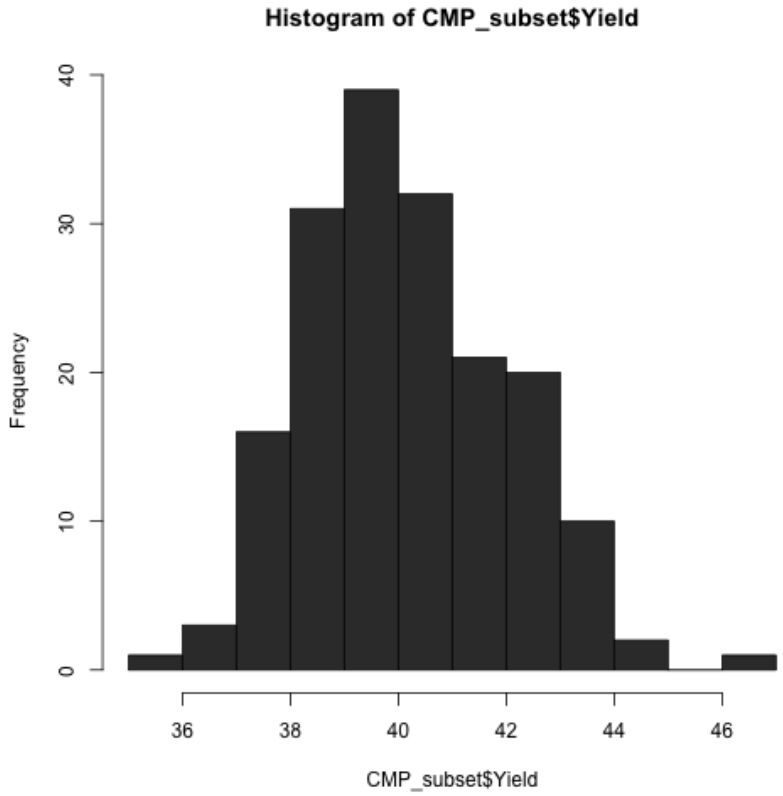
Graphical summary

```
# Univariate plot: histogram.
hist(CMP_subset$Yield)
```

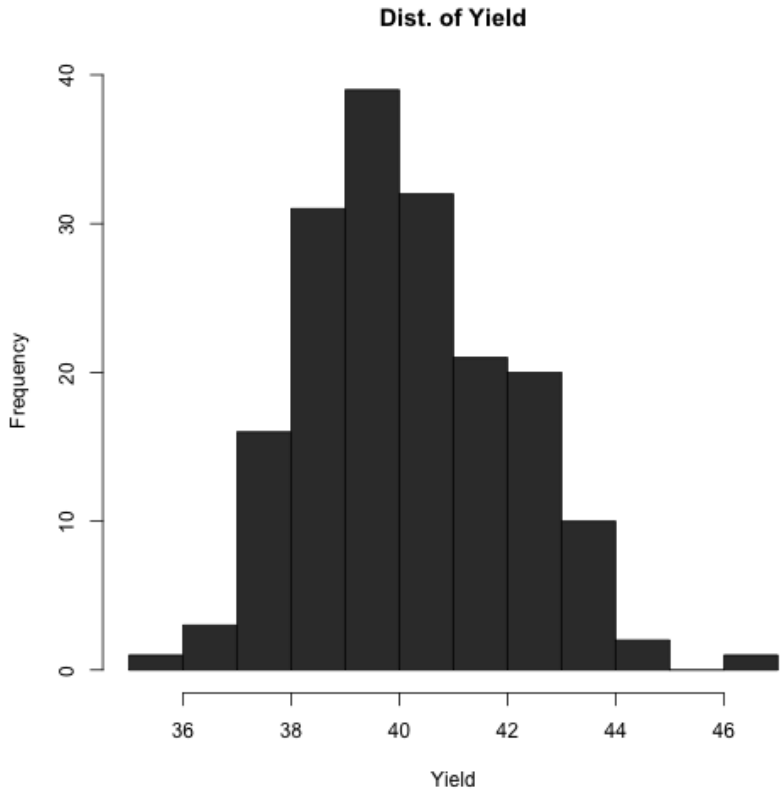


Univariate plots: histogram

```
# Customize your histogram.
hist(CMP_subset$Yield,
     col = col_sample[1]) #<- set color
```

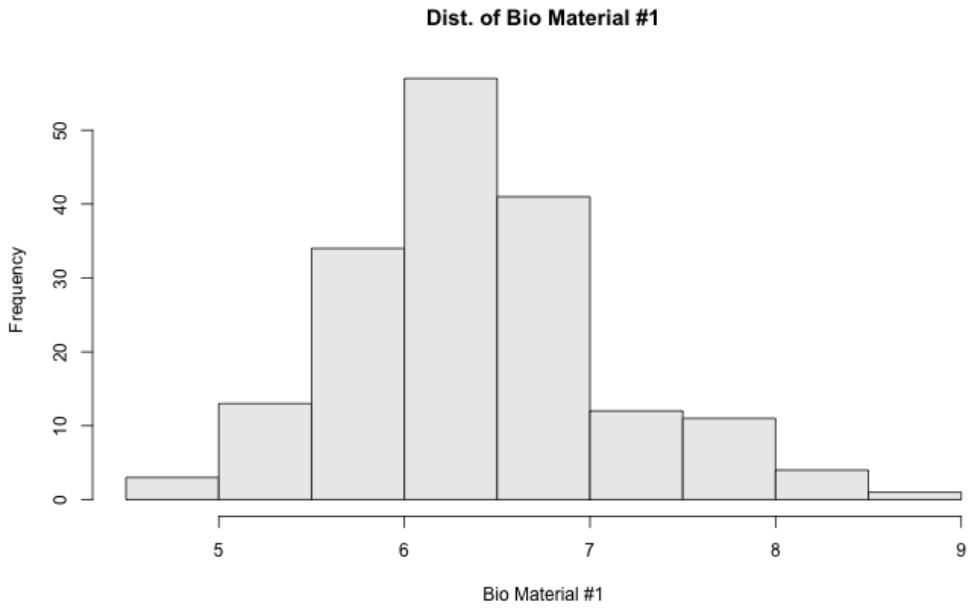


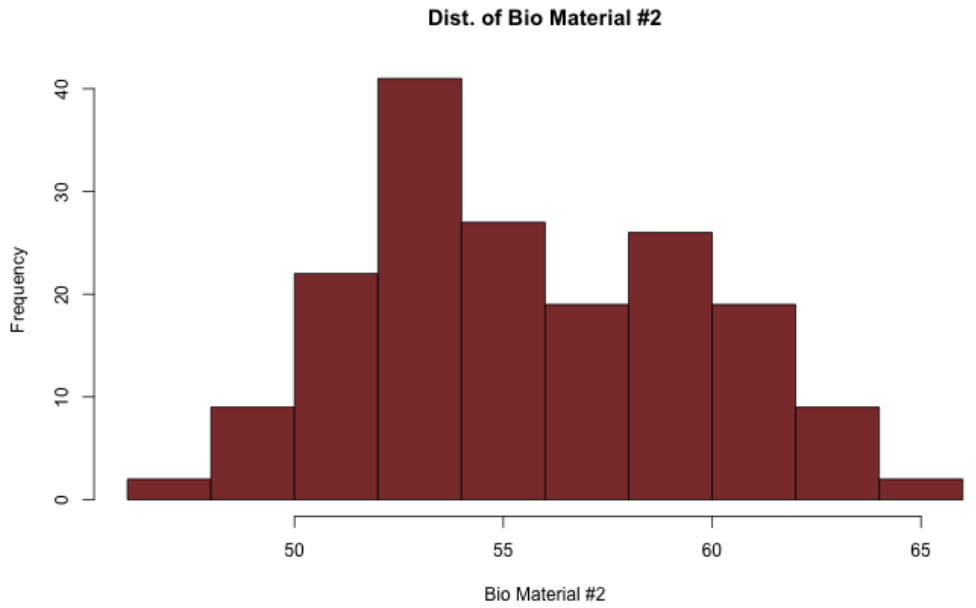

```
# Customize your histogram.
hist(CMP_subset$Yield,
     col = col_sample[1],
     xlab = "Yield",   #<- set x-axis label
     main = "Dist. of Yield") #<- set title
```



Compare variables: histograms combined

```
# Make a combined histogram plot.
par(                                    #<- set plot area parameters with `par`
    mfrow = c(1, 2))                    #<- split area into 1 row & 2 columns with `mfrow`
hist(CMP_subset$BiologicalMaterial01,   #<- add 1st histogram
     col = col_sample[2],
     xlab = "Bio Material #1",
     main = "Dist. of Bio Material #1")
hist(CMP_subset$BiologicalMaterial02,   #<- add 2nd histogram
     col = col_sample[3],
     xlab = "Bio Material #2",
     main = "Dist. of Bio Material #2")
```





Knowledge check 1

Exercise 1

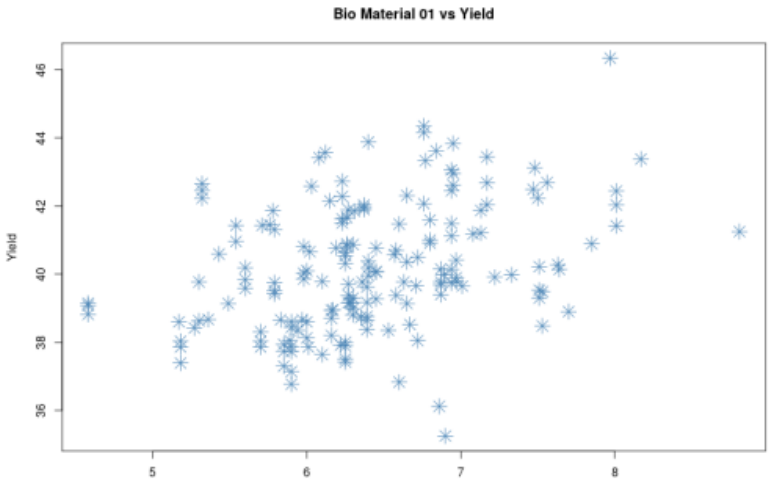
Module completion checklist

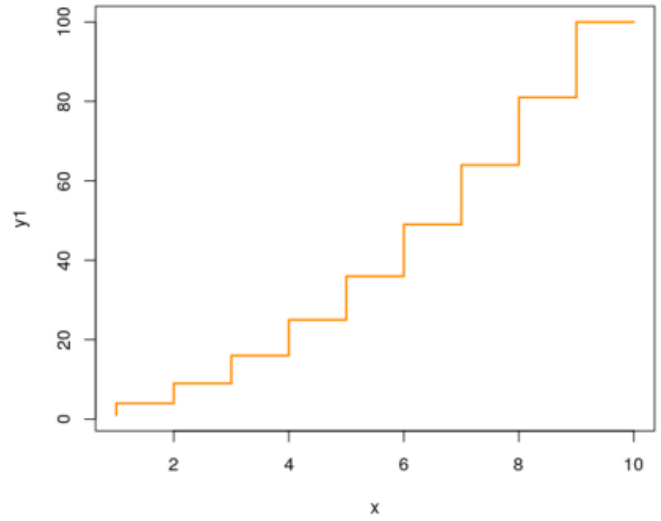
Objective	Complete
Define exploratory data analysis (EDA) and its cycle	✔
Define and differentiate between static and interactive visualizations	✔
Choose when to use univariate plots to illustrate patterns in data	✔
Construct univariate plots using base R	✔
Choose when to use bivariate plots to illustrate patterns in data	
Construct bivariate plot using base R	
Choose when to use multivariate plots to illustrate patterns in data	
Construct multivariate plot (scatterplot matrix) using base R	
Measure and observe correlation using corrplot package	
Formulate the process of using ggplot2 to build complex plots	
Build univariate plots using various geom layers with ggplot2	
Build bivariate plots using various geom layers with ggplot2	

Bivariate plots

Use cases

- Bivariate plots are used to visualize data distribution and relationships between **two variables**
- They are used heavily throughout different stages of EDA **to learn more about how one variable is related to another**
- They are also used **in combination with other bivariate plots to compare relationships between different pairs of variables**
- Bivariate plots include the following popular graphs: `scatterplot`, `line graph`





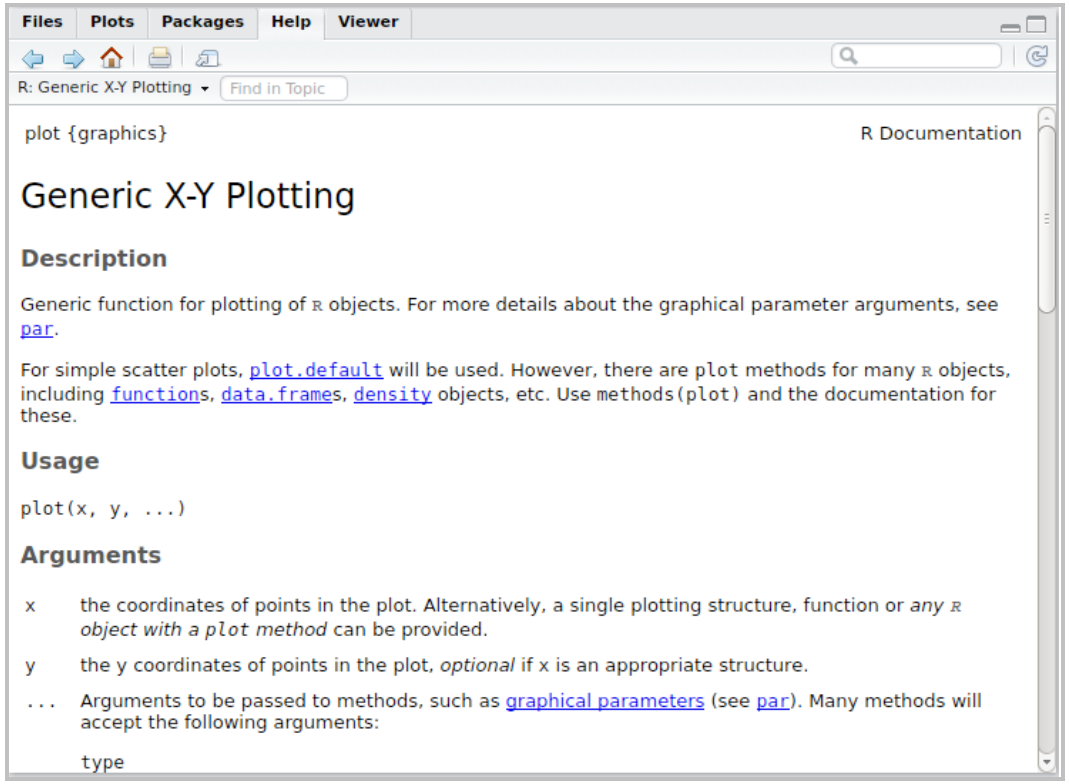
Module completion checklist

Objective	Complete
Define exploratory data analysis (EDA) and its cycle	✔
Define and differentiate between static and interactive visualizations	✔
Choose when to use univariate plots to illustrate patterns in data	✔
Construct univariate plots using base R	✔
Choose when to use bivariate plots to illustrate patterns in data	✔
Construct bivariate plot using base R	
Choose when to use multivariate plots to illustrate patterns in data	
Construct multivariate plot (scatterplot matrix) using base R	
Measure and observe correlation using corrplot package	
Formulate the process of using ggplot2 to build complex plots	
Build univariate plots using various geom layers with ggplot2	
Build bivariate plots using various geom layers with ggplot2	

Bivariate plots: R `plot` function

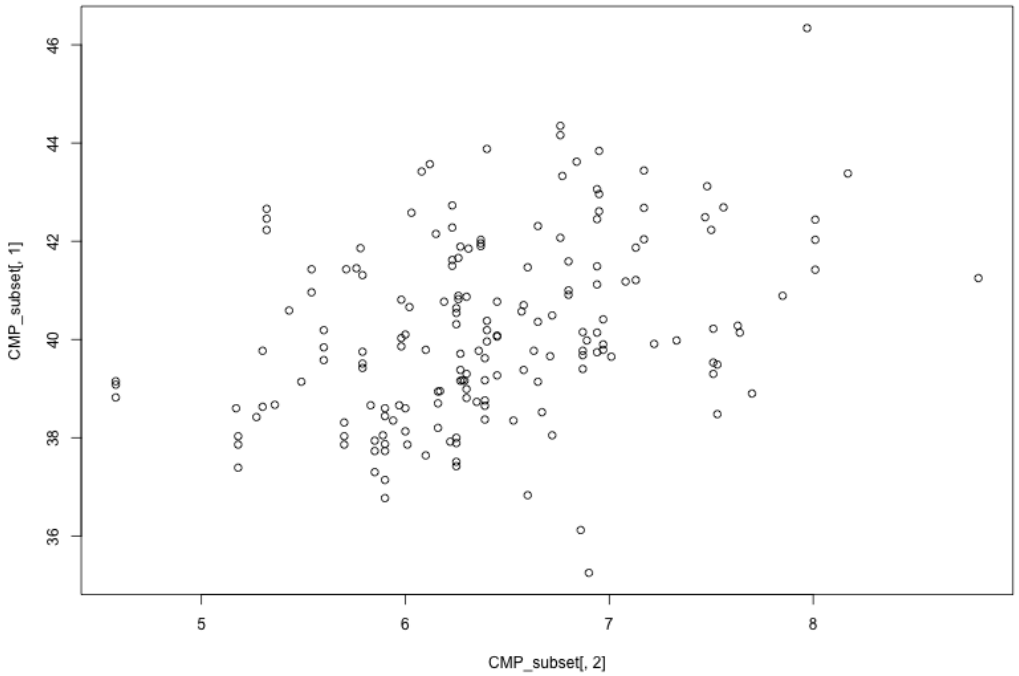
```
?plot
```

- The most popular, robust, and useful visualization of numeric data in R
- It takes 2 main arguments:
  - Variable for `x-axis`, and
  - Variable for `y-axis`
- Produces a basic **scatterplot**
- Can also produce a **lineplot** with certain combination of parameters

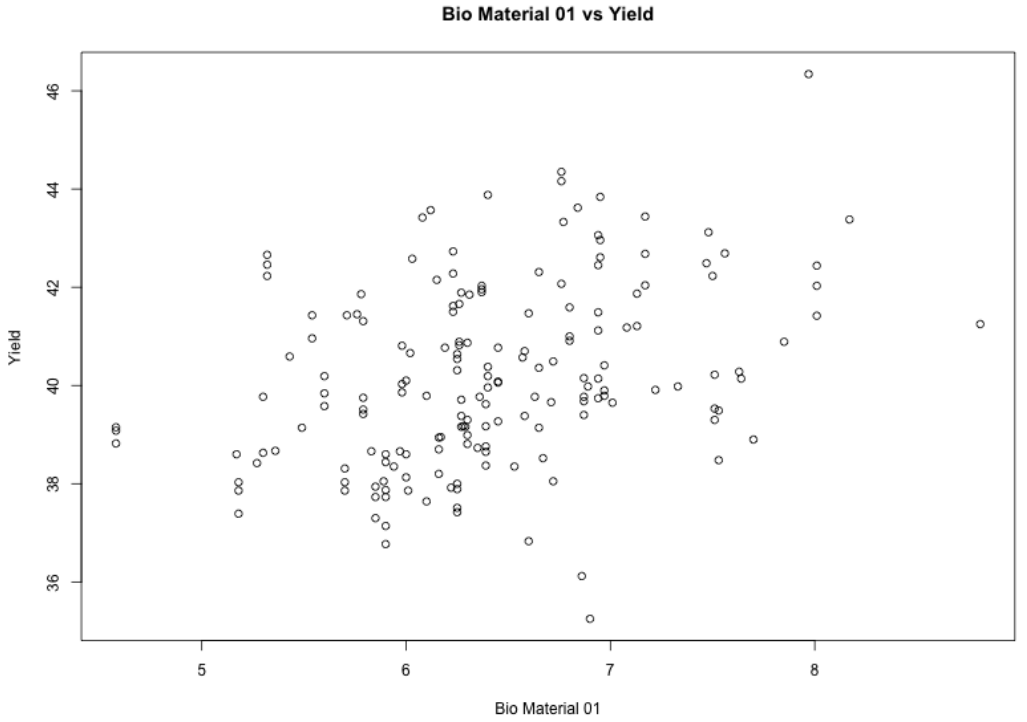


Bivariate plots: scatterplot

```
# Generic scatterplot.
plot(CMP_subset[ , 2], #<- variable for x-axis
     CMP_subset[ , 1]) #<- variable for y-axis
```

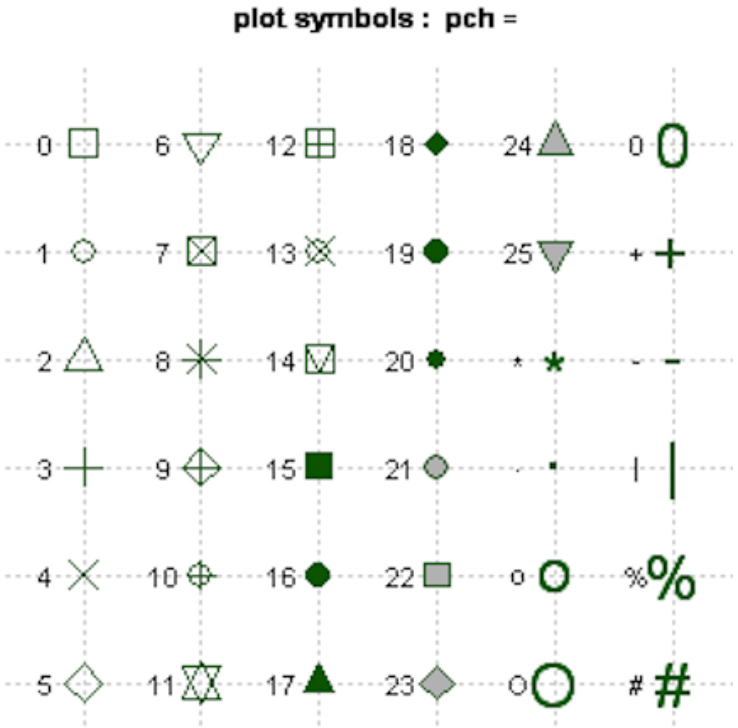



```
# Add axis labels and title.
plot(CMP_subset[ , 2],
     CMP_subset[ , 1],
     xlab = "Bio Material 01",
     ylab = "Yield",
     main = "Bio Material 01 vs Yield")
```

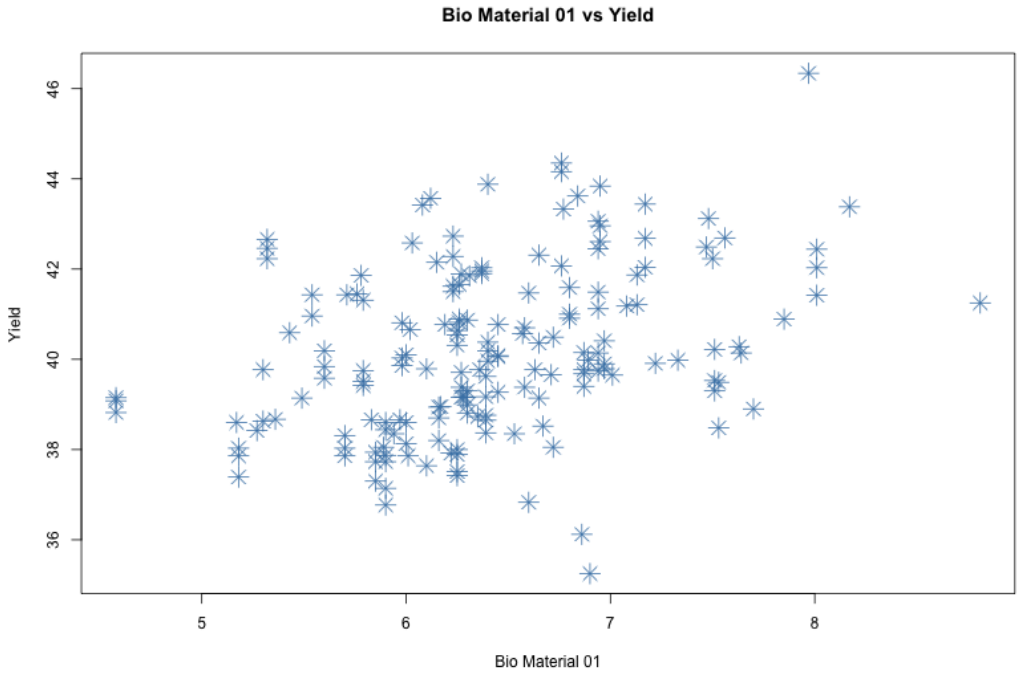


Bivariate plots: scatterplot

- Select a symbol with `pch` option from the following




```
# Customize your scatterplot.
plot(CMP_subset[ , 2],
     CMP_subset[ , 1],
     xlab = "Bio Material 01",
     ylab = "Yield",
     main = "Bio Material 01 vs Yield",
     pch = 8,   #<- type of symbol to use
     cex = 2,   #<- scale of the point
     col = "steelblue")
```



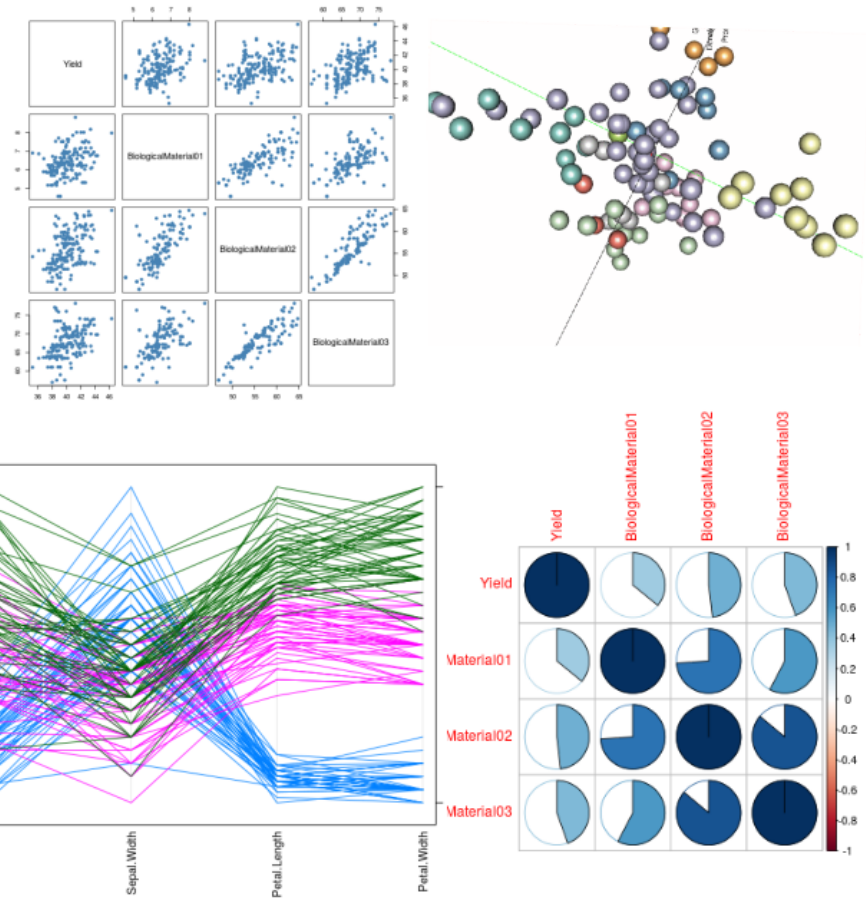
Module completion checklist

Objective	Complete
Define exploratory data analysis (EDA) and its cycle	✔
Define and differentiate between static and interactive visualizations	✔
Choose when to use univariate plots to illustrate patterns in data	✔
Construct univariate plots using base R	✔
Choose when to use bivariate plots to illustrate patterns in data	✔
Construct bivariate plot using base R	✔
Choose when to use multivariate plots to illustrate patterns in data	
Construct multivariate plot (scatterplot matrix) using base R	
Measure and observe correlation using corrplot package	
Formulate the process of using ggplot2 to build complex plots	
Build univariate plots using various geom layers with ggplot2	
Build bivariate plots using various geom layers with ggplot2	

Multivariate plots

Use cases

- Multivariate plots are used to visualize data distribution and relationships between **3 or more variables** (3D plots are essentially a special case of multivariate plots)
- They are used extensively during different stages of EDA and in combination with machine learning methods **to learn more about how variables are related to each other**
- Multivariate plots include the following popular graphs: `scatterplot matrix`, `correlation plot`, `parallel coordinates plot`, various compound plots using multiple juxtaposed variables



Knowledge check 2

Exercise 2

Module completion checklist

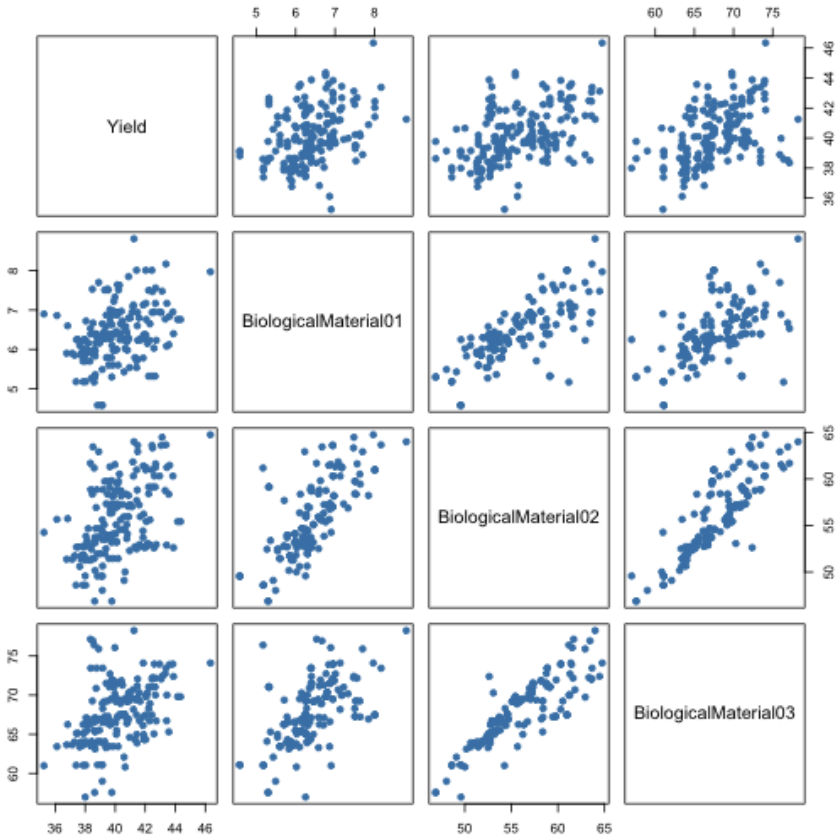
Objective	Complete
Define exploratory data analysis (EDA) and its cycle	✔
Define and differentiate between static and interactive visualizations	✔
Choose when to use univariate plots to illustrate patterns in data	✔
Construct univariate plots using base R	✔
Choose when to use bivariate plots to illustrate patterns in data	✔
Construct bivariate plot using base R	✔
Choose when to use multivariate plots to illustrate patterns in data	✔
Construct multivariate plot (scatterplot matrix) using base R	
Measure and observe correlation using corrplot package	
Formulate the process of using ggplot2 to build complex plots	
Build univariate plots using various geom layers with ggplot2	
Build bivariate plots using various geom layers with ggplot2	

Multivariate plots: scatterplot matrix

- If you would like to quickly glance at relationships between multiple variables, **scatterplot matrix** is your friend
- Use `pairs` function, pass a range of variables to the plotting function and voila!

Scatterplot matrix is symmetric around the diagonal, each variable occupies the same row and column, to see the scatterplot for variable 1 and 2, pick the tile in row 1 and column 2!

```
# Plot scatterplots for many variables.
pairs(CMP_subset[, 1:4], #<- select variables
      pch = 19,          #<- set symbol
      col = "steelblue") #<- set color
```



Module completion checklist

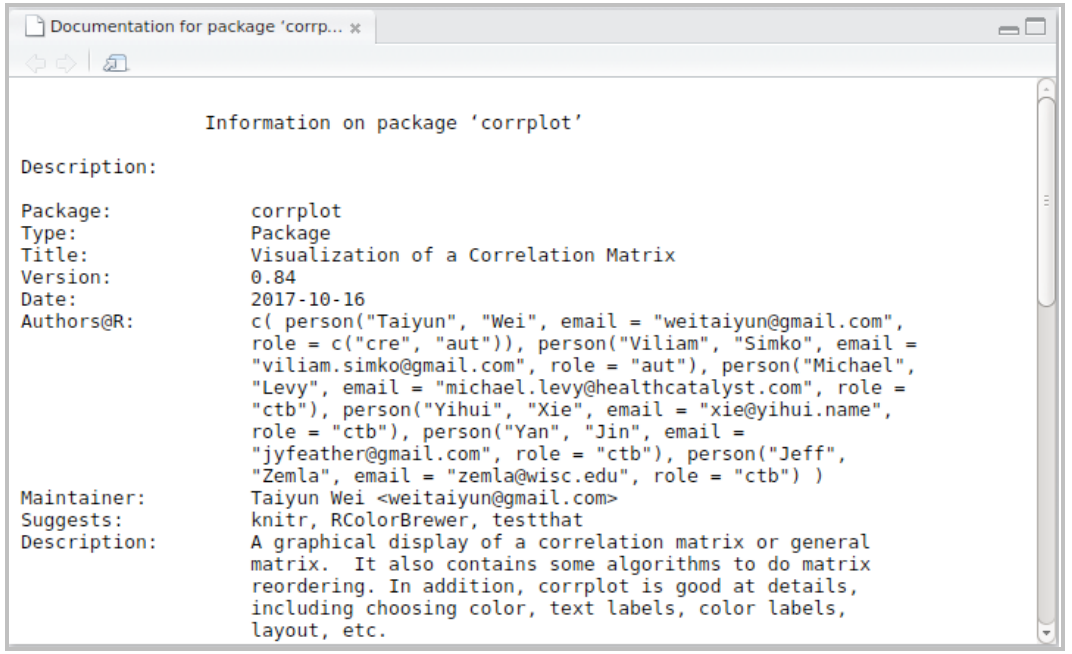
Objective	Complete
Define exploratory data analysis (EDA) and its cycle	✔
Define and differentiate between static and interactive visualizations	✔
Choose when to use univariate plots to illustrate patterns in data	✔
Construct univariate plots using base R	✔
Choose when to use bivariate plots to illustrate patterns in data	✔
Construct bivariate plot using base R	✔
Choose when to use multivariate plots to illustrate patterns in data	✔
Construct multivariate plot (scatterplot matrix) using base R	✔
Measure and observe correlation using corrplot package	
Formulate the process of using ggplot2 to build complex plots	
Build univariate plots using various geom layers with ggplot2	
Build bivariate plots using various geom layers with ggplot2	

Multivariate plots: correlation plot

- A **scatterplot matrix** is useful because we can quickly glance and see if the distribution of points follows a distinct pattern
- A **correlation plot** is also a multivariate plot in a form of a matrix, but instead of showing the point distribution, it displays correlation between variables
- The `corrplot` package allows us to plot it

Installing `corrplot` package

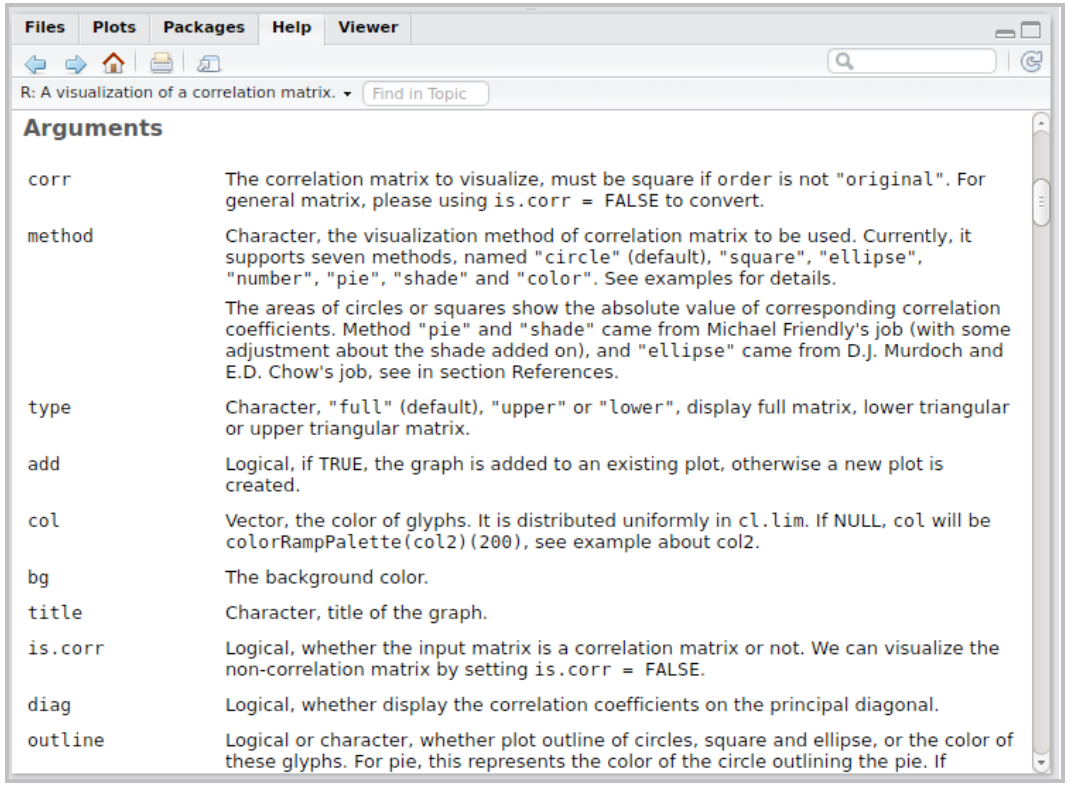
```
# Install package through command line.
install.packages("corrplot")

# Load the package into the environment.
library(corrplot)

# View package documentation.
library(help = "corrplot")
```



Multivariate plots: correlation plot

```
?corrplot
```

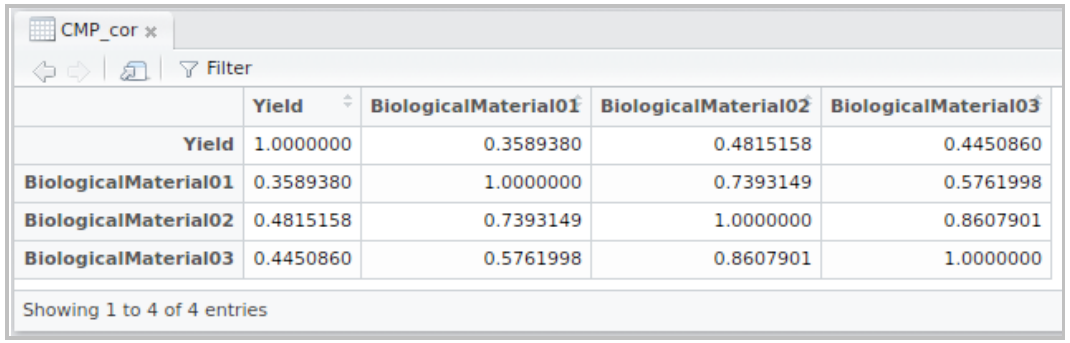
- A correlation plot requires as its first (and only) mandatory argument a **correlation matrix** of the variables we would like to show in our plot



Computing correlations between variables

```
# Compute a correlation matrix of first 4
# variables using `cor` function.
CMP_cor = cor(CMP_subset[, 1:4])
```

```
View(CMP_cor)
```



CMP_cor

Filter

	Yield	BiologicalMaterial01	BiologicalMaterial02	BiologicalMaterial03
Yield	1.0000000	0.3589380	0.4815158	0.4450860
BiologicalMaterial01	0.3589380	1.0000000	0.7393149	0.5761998
BiologicalMaterial02	0.4815158	0.7393149	1.0000000	0.8607901
BiologicalMaterial03	0.4450860	0.5761998	0.8607901	1.0000000

Showing 1 to 4 of 4 entries

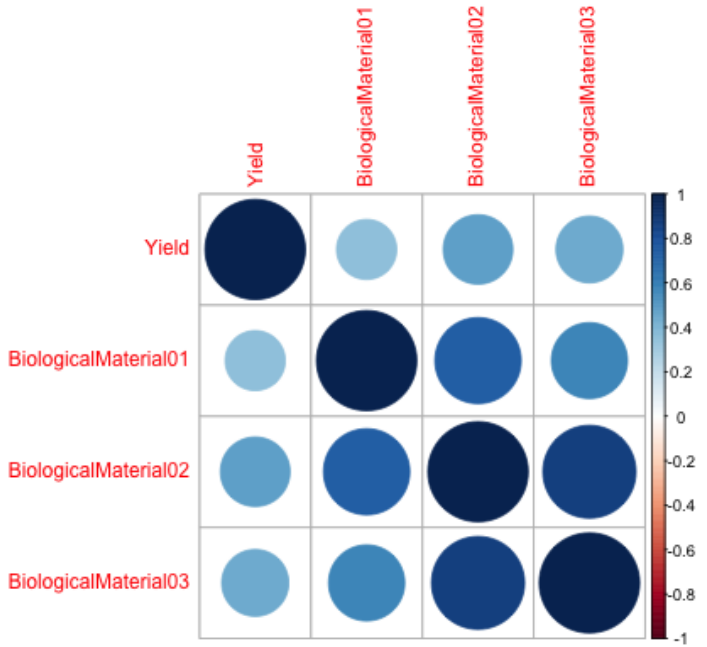
A correlation matrix has a few important properties:

1. **It is square** (i.e. the number of rows is the same as the number of columns)
2. **It is symmetric** (i.e. values on opposite sides of the diagonal are mirrored $value_{row_i,col_j} = value_{row_j,col_i}$)
3. Values on the **diagonal are equal to 1**
4. Each value in the matrix is a **correlation coefficient**, which is a value between $[-1, 1]$

Computing correlations between variables

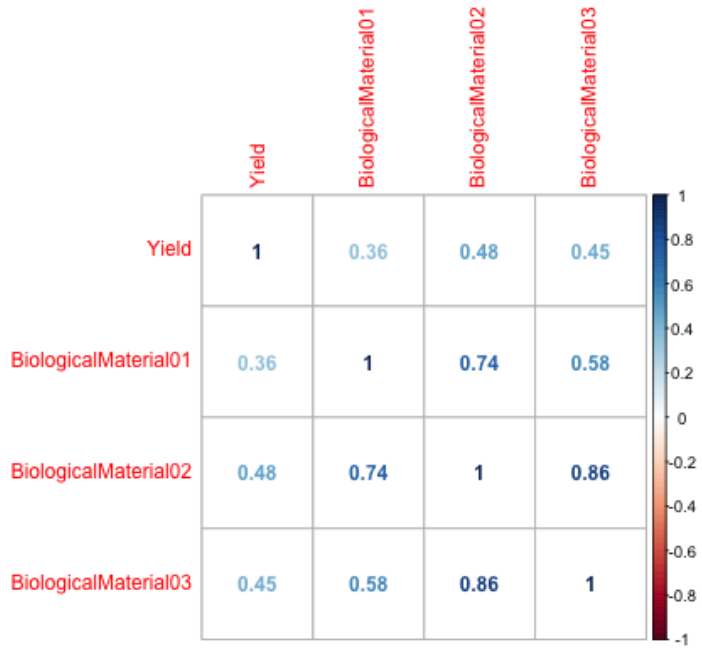
- The size of a circle represents the absolute value of the corr. coefficient:
  $0 \leq |cor| \leq 1$
- The color of a circle represents the `sign` (i.e. **positive** vs. **negative**)
- From this plot alone, we notice that:
  - All variables are **positively** correlated
  - `BiologicalMaterial01` & `BiologicalMaterial02` have a high corr. coefficient of about `0.75` and `BiologicalMaterial02` & `BiologicalMaterial03` have an even higher coefficient of over `0.8`!

```
# Create correlation plot.
corrplot(CMP_cor)
```

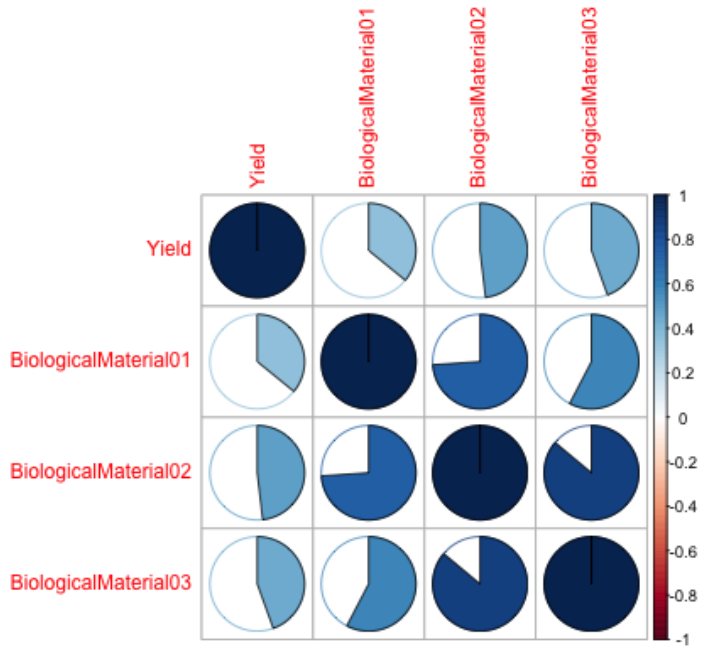


Multivariate plots: correlation plot

```
# The default method is "circle", to
# display the correlation coefficient
# instead, use method "number".
corrplot(CMP_cor, method = "number")
```




```
# To display correlation coefficient as
# a portion of a circle proportionate to
# correlation coefficient, use method "pie".
corrplot(CMP_cor, method = "pie")
```

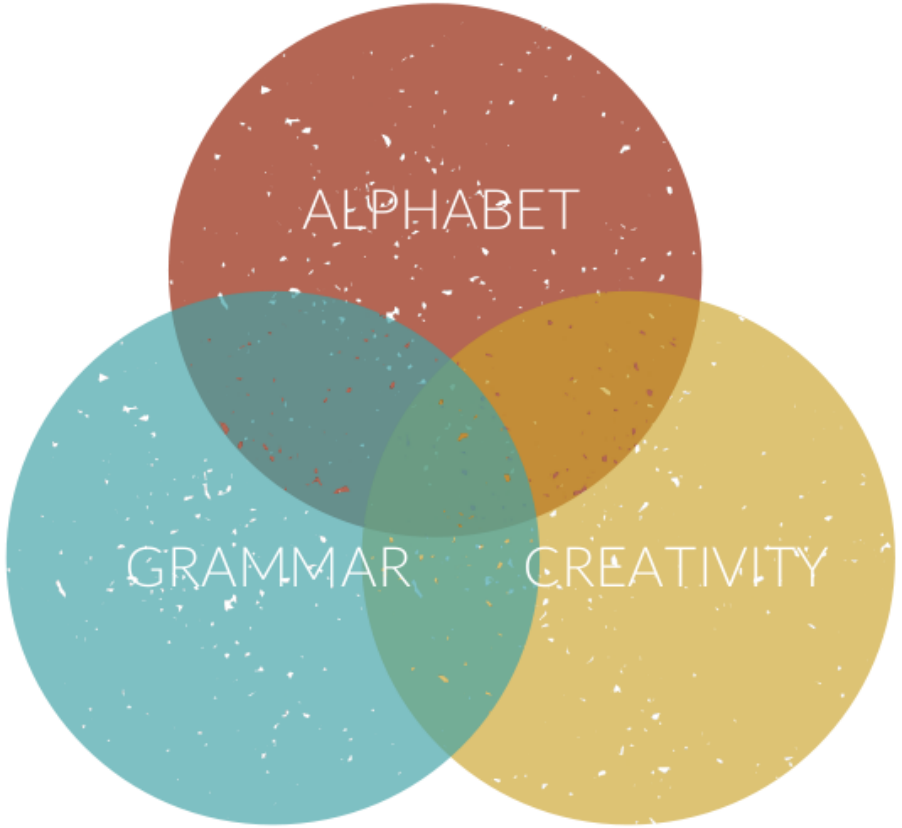


Module completion checklist

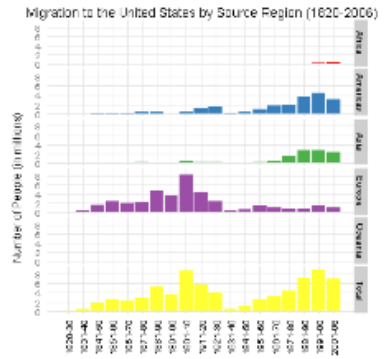
Objective	Complete
Define exploratory data analysis (EDA) and its cycle	✔
Define and differentiate between static and interactive visualizations	✔
Choose when to use univariate plots to illustrate patterns in data	✔
Construct univariate plots using base R	✔
Choose when to use bivariate plots to illustrate patterns in data	✔
Construct bivariate plot using base R	✔
Choose when to use multivariate plots to illustrate patterns in data	✔
Construct multivariate plot (scatterplot matrix) using base R	✔
Measure and observe correlation using corrplot package	✔
Formulate the process of using ggplot2 to build complex plots	
Build univariate plots using various geom layers with ggplot2	
Build bivariate plots using various geom layers with ggplot2	

Visualizing data with `ggplot2` package

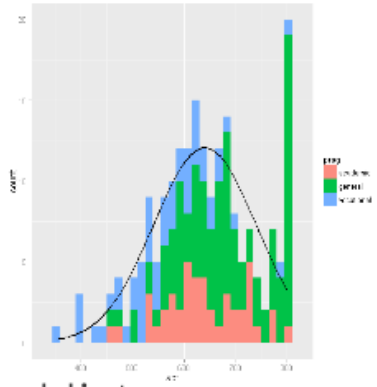
- Explore your data efficiently
- Communicate a visual story flexibly and efficiently
- Layer raw, summarized and contextual data
  - demonstrate relationships
- Reproduce and extend your work easily



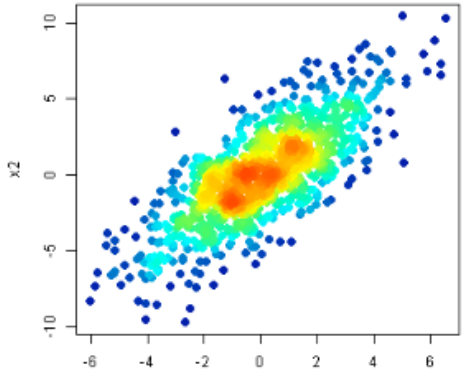
Visualizing data with `ggplot2` package



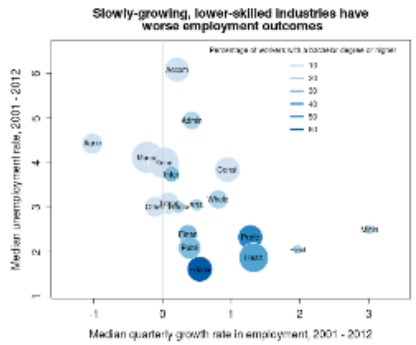
Bar plot



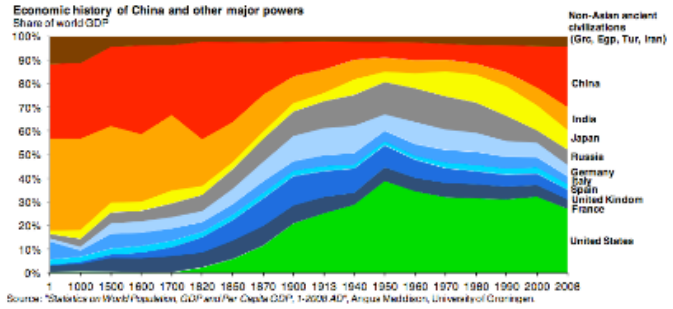
Histogram



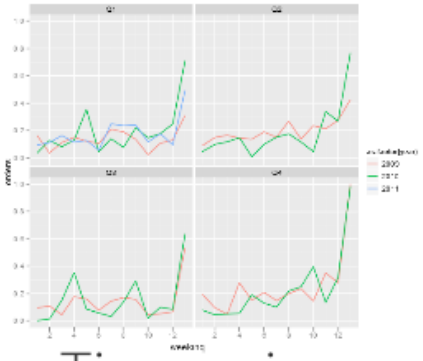
Scatterplot



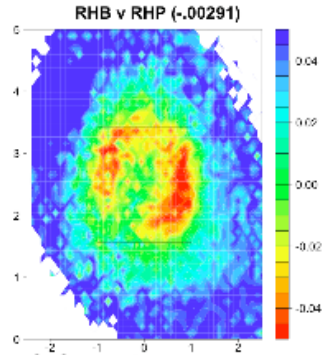
Bubble plot



Area plot



Time series



Heat map

Working with `ggplot2` package

Set up

1. Specify data
2. Link data to visuals
3. Assign shapes

Adjust

1. Visual effects
2. Axes
3. Legend

Polish

1. Customize theme
2. Layer statistics
3. Text

Knowledge check 3

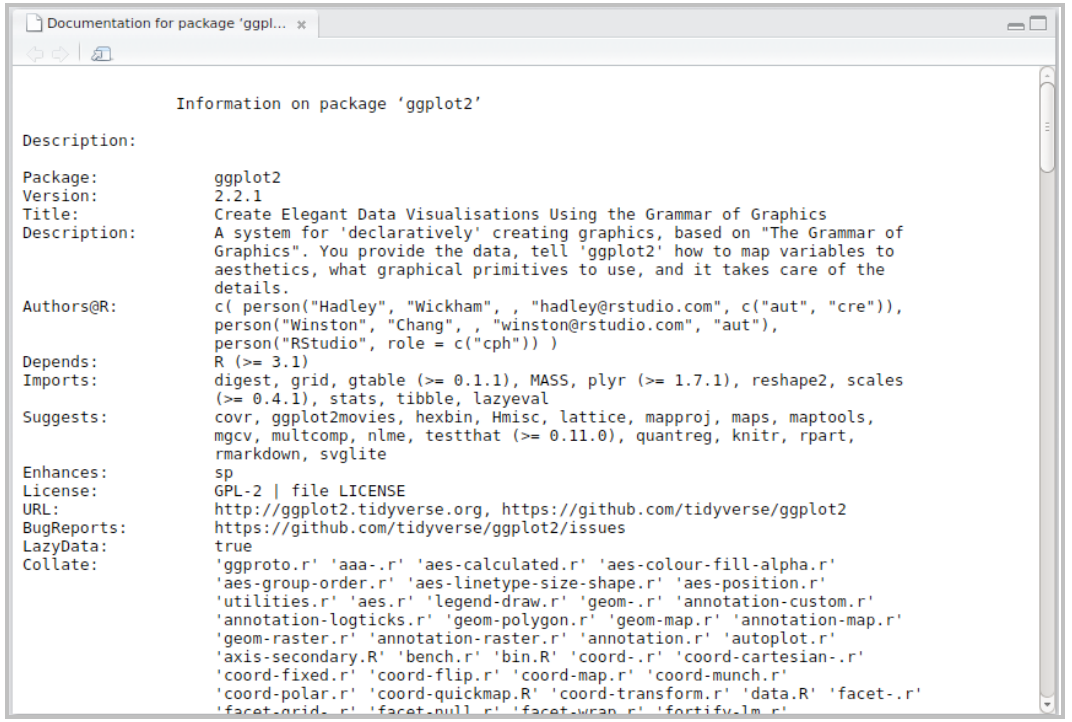
Exercise 3

Module completion checklist

Objective	Complete
Define exploratory data analysis (EDA) and its cycle	✔
Define and differentiate between static and interactive visualizations	✔
Choose when to use univariate plots to illustrate patterns in data	✔
Construct univariate plots using base R	✔
Choose when to use bivariate plots to illustrate patterns in data	✔
Construct bivariate plot using base R	✔
Choose when to use multivariate plots to illustrate patterns in data	✔
Construct multivariate plot (scatterplot matrix) using base R	✔
Measure and observe correlation using corrplot package	✔
Formulate the process of using ggplot2 to build complex plots	✔
Build univariate plots using various geom layers with ggplot2	
Build bivariate plots using various geom layers with ggplot2	

Working with `ggplot2` package

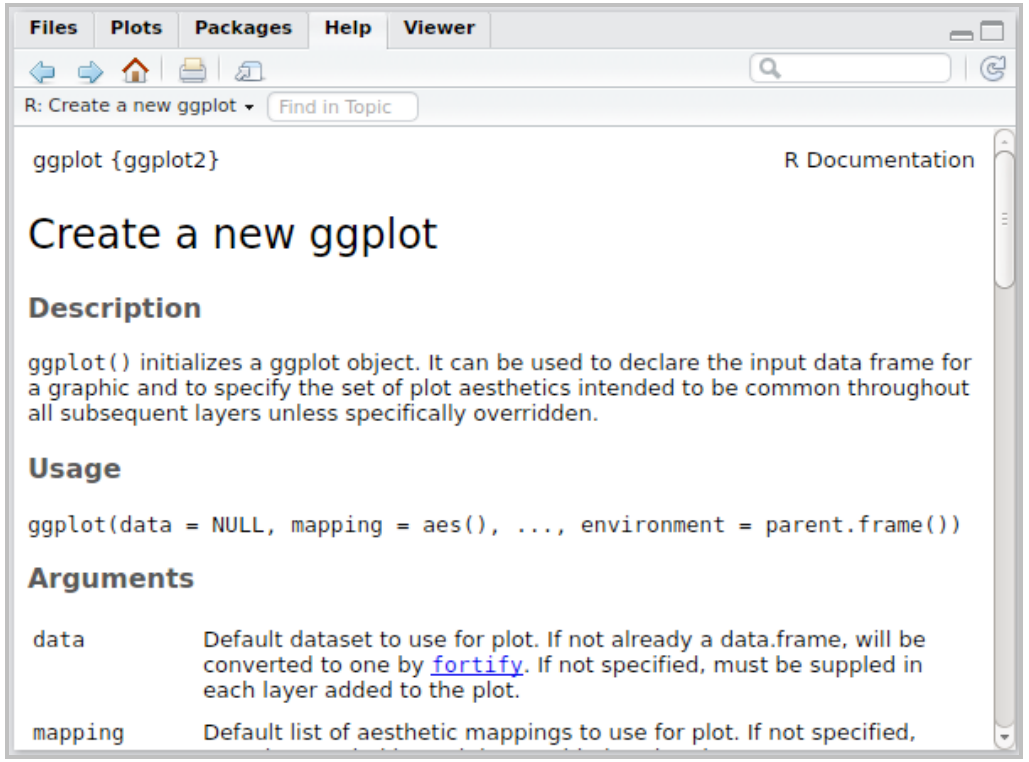
```
# Since `ggplot2` is an external package,
# we need to install it first.
install.packages("ggplot2")

# Then, we need to load it to our environment.
library(ggplot2)

# Take a look at the documentation.
library(help = "ggplot2")
```



Working with `ggplot2` package

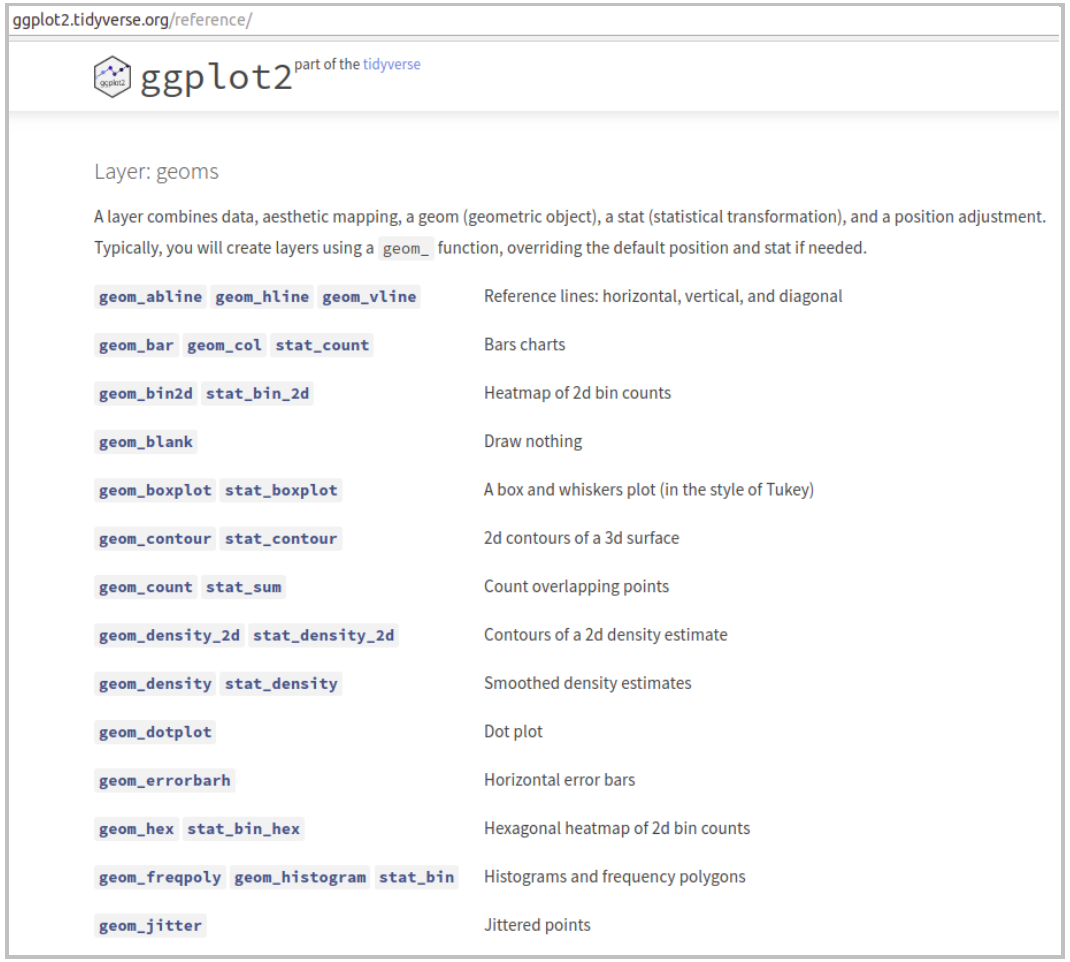
```
?ggplot
```

1. The main function in the package is `ggplot`, which creates the initial plot object (i.e. an empty canvas)
2. The main 2 arguments in the function are
   i. `data`: lets `ggplot` know which data is to be plotted
   ii. `mapping`: a named list of `aes[thetics]` that lets `ggplot` know which variables should be mapped to which axes



Working with `ggplot2` package

- Each chart in `ggplot2` package is like a layered cake
- After the initial plot object has been created as a base, other layers are put on top of it
- Each chart type corresponds to a `geom` (i.e. layer), some of the most used `geoms` in the R community are: `geom_histogram`, `geom_point`, `geom_smooth`, and `geom_density`
- You can view the entire list of `geoms` on ***http://ggplot2.tidyverse.org/reference/*** along with all documentation, help pages, vignettes, and code/chart examples



First stage of plotting with `ggplot2`: setup

In order to make a base plot, we need to:

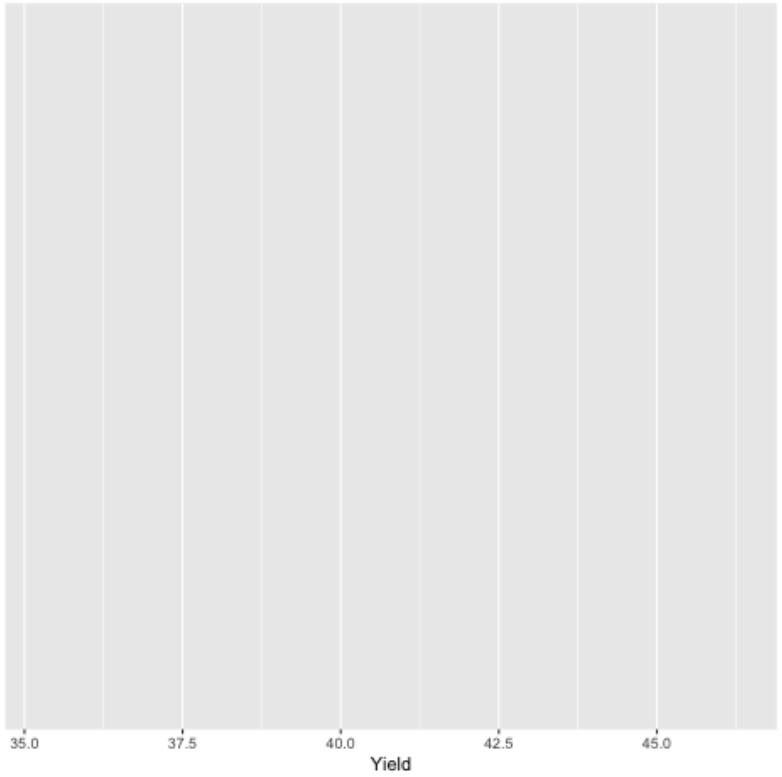
1. Specify data
2. Link data to visuals
3. Assign shapes

Translated into `ggplot2` syntax, this will require to call a `ggplot` function and

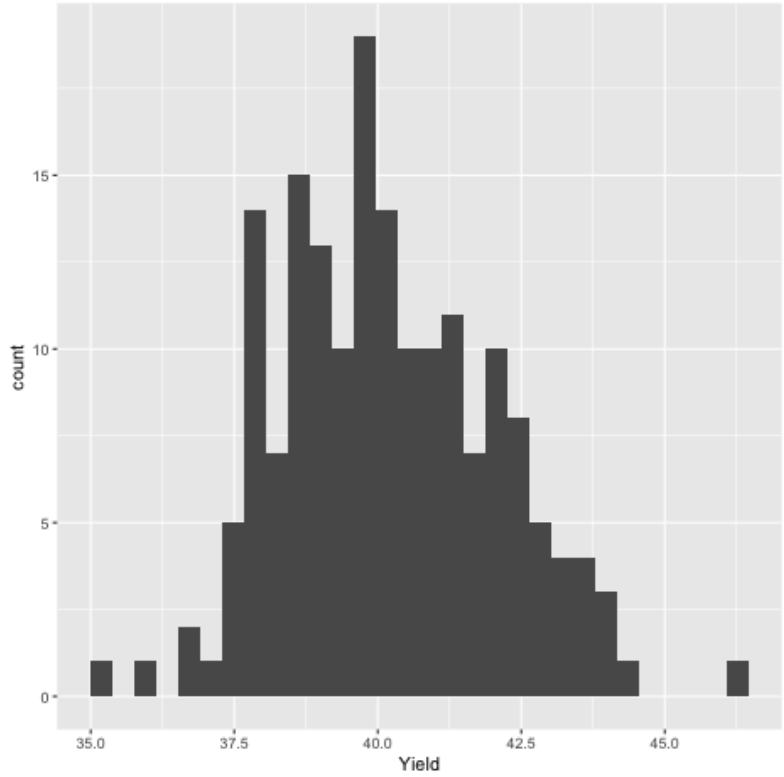
1. Set the first argument to the data we want to use (e.g. `CMP_subset`)
2. Set the second argument to the mapping of our choice (e.g. `Yield` to be plotted on `x-axis`)
3. Add a `geom` layer to produce a histogram (e.g. add `geom_histogram` to the base plot)

Making a plot with `geom_histogram`

```
# Let's create a base plot.
ggp1 = ggplot(CMP_subset,      #<- set data
              aes(x = Yield)) #<- set aesthetics
ggp1
```

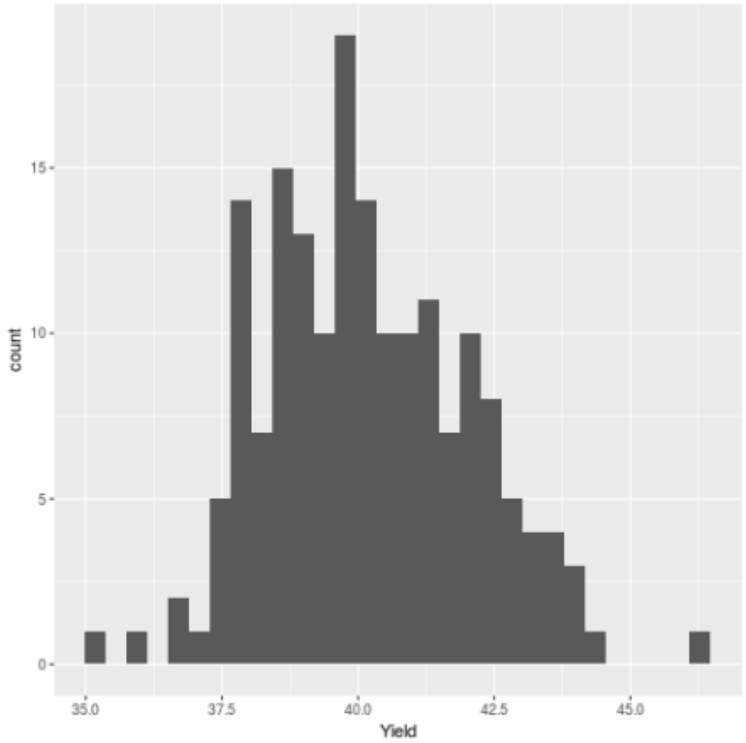



```
# Let's add a layer.
# Each layer is called a `geom` - we need to use
# `geom_histogram` to add a histogram layer.
ggp1 + geom_histogram()
```

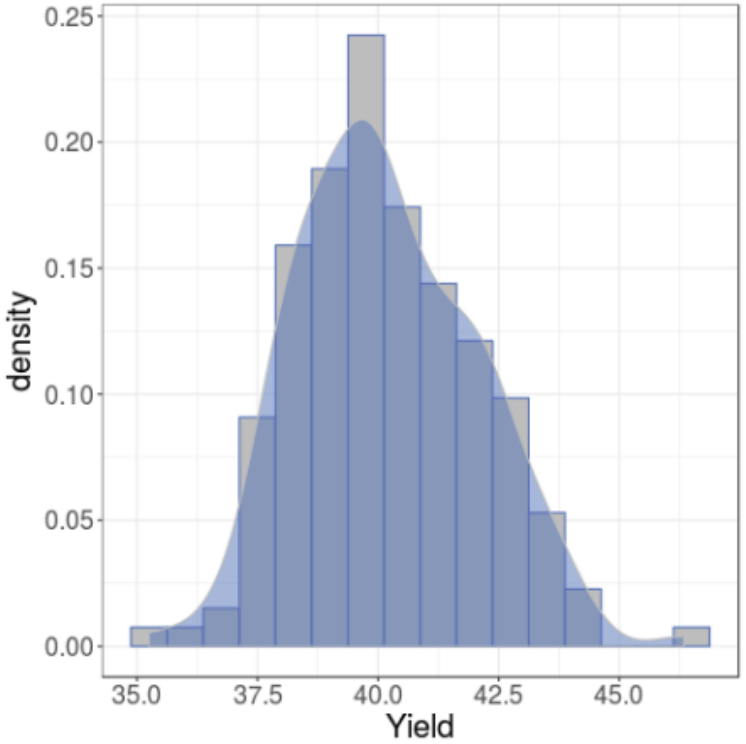


Building a histogram with density with `ggplot2`

What we have



What we need



Second stage of plotting with `ggplot2`: adjust

To make the visualization prominent, adjust:

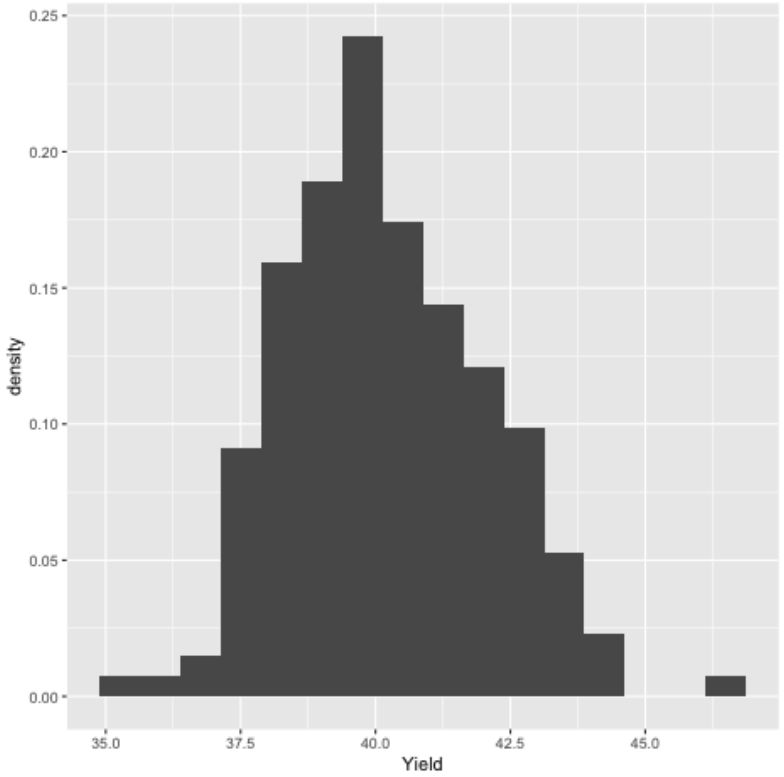
1. Vis. effects
2. Axes
3. Legend

Depending on the plot and the final goal, you may want to

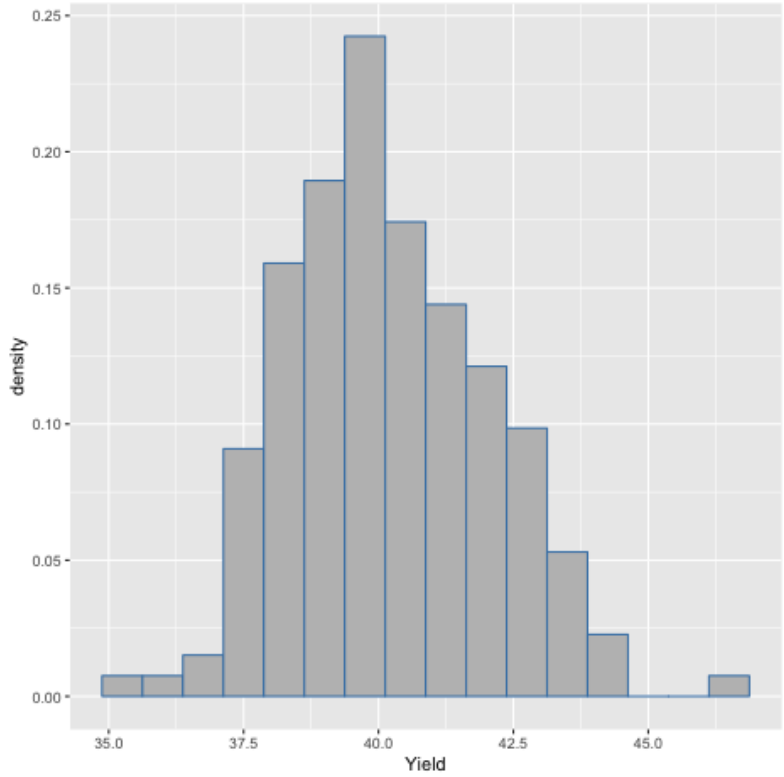
1. Change colors, add more layers with extra information (i.e. additional data sources, reference lines or points etc.)
2. Add axes labels and adjust the breaks in data (i.e. change scale, bins in the histogram)
3. Add a legend if there are more variables or datasets than one combined on the plot

Making a plot with `geom_histogram`: adjust

```
# Say, instead of frequency counts, we want
# probability density estimates. We can:
ggp1 +
  # 1. Make `y-axis` of type `density`.
  geom_histogram(aes(y = ..density..),
  # 2. Adjust binwidth for better smoothness.
                 binwidth = 0.75)
```




```
ggp1 = ggp1 +    #<- save adjusted plot
  geom_histogram(aes(y = ..density..),
                 binwidth = 0.75,
                 # 3. Add color & fill.
                 color = "steelblue",
                 fill = "gray")
ggp1             #<- view saved plot
```



Third stage of plotting with `ggplot2`: polish

In order to make visualization complete, we need to:

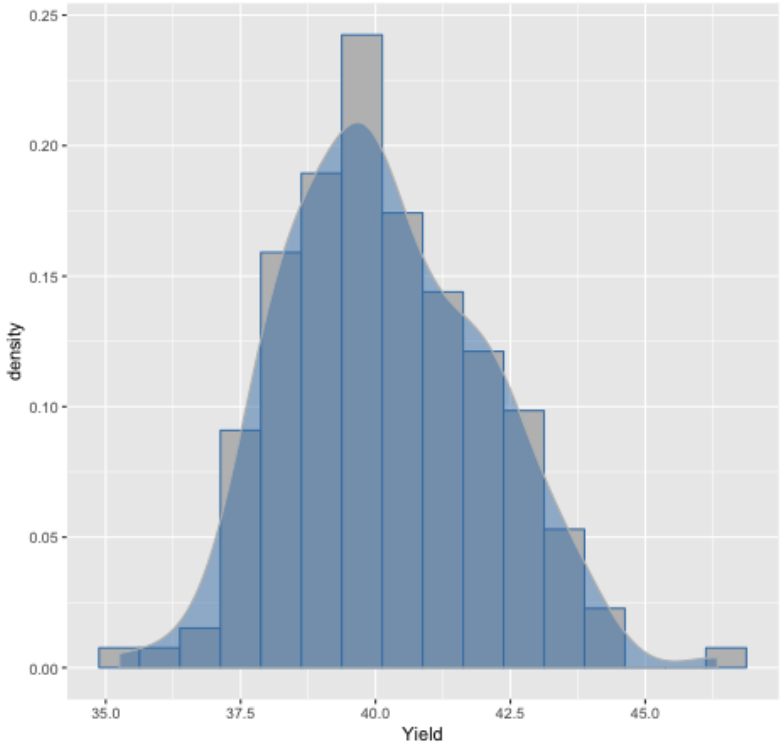
1. Layer statistics
2. Customize theme
3. Text

Depending on the plot and the final goal, you may want to

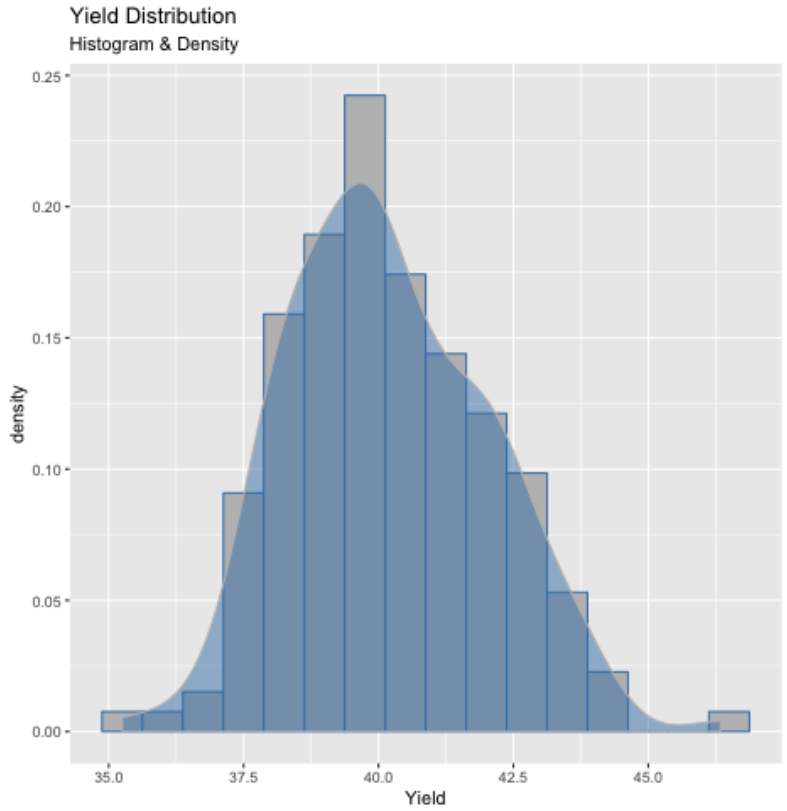
1. Add statistical layers (i.e. density, mean markers, LOESS curves, etc.)
2. Customize plot theme
3. Add/change any text labels or add free-form text to the plot

Making a plot with `geom_histogram`: polish

```
ggp1 = ggp1 +
  # Add density layer with 50% opaque fill
  # in "steelblue" and gray color (border).
  geom_density(alpha = .5,
               color = "gray",
               fill = "steelblue")
ggp1
```



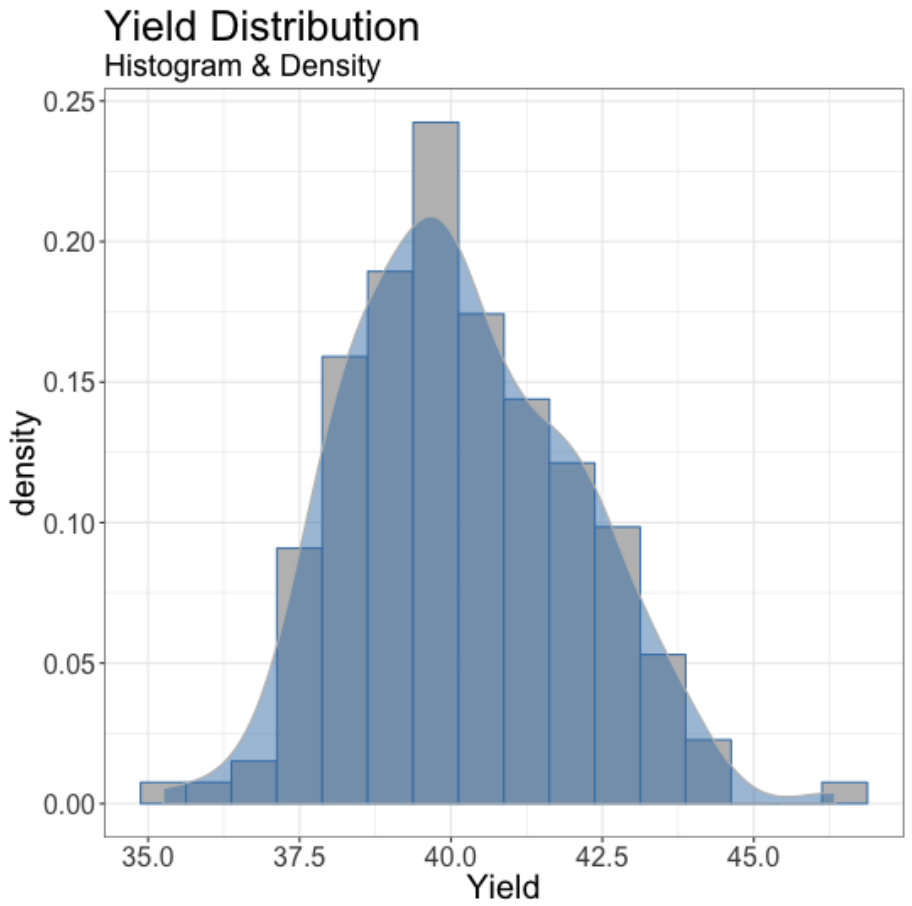

```
ggp1  = ggp1 +
# Add plot title and subtitle.
labs(title = "Yield Distribution",
     subtitle = "Histogram & Density")
ggp1
```



Making a plot with `geom_histogram`: polish

```
# Add a black and white theme to
# overwrite default.
ggp1 = ggp1 +

  # Add a black and white theme.
  theme_bw() +

  # Customize elements of the theme.
  theme(axis.title = element_text(size = 20),
        axis.text = element_text(size = 16),
        plot.title = element_text(size = 25),
        plot.subtitle = element_text(size = 18))
```

```
# Display polished plot.
ggp1
```



Module completion checklist

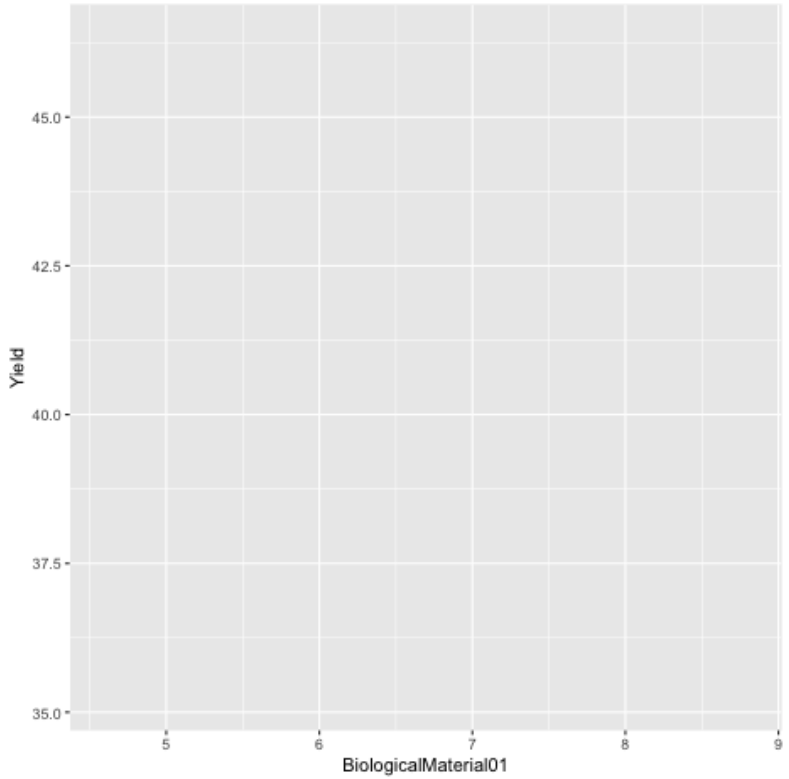
Objective	Complete
Define exploratory data analysis (EDA) and its cycle	✔
Define and differentiate between static and interactive visualizations	✔
Choose when to use univariate plots to illustrate patterns in data	✔
Construct univariate plots using base R	✔
Choose when to use bivariate plots to illustrate patterns in data	✔
Construct bivariate plot using base R	✔
Choose when to use multivariate plots to illustrate patterns in data	✔
Construct multivariate plot (scatterplot matrix) using base R	✔
Measure and observe correlation using corrplot package	✔
Formulate the process of using ggplot2 to build complex plots	✔
Build univariate plots using various geom layers with ggplot2	✔
Build bivariate plots using various geom layers with ggplot2	

Making a scatterplot with `geom_point`: setup

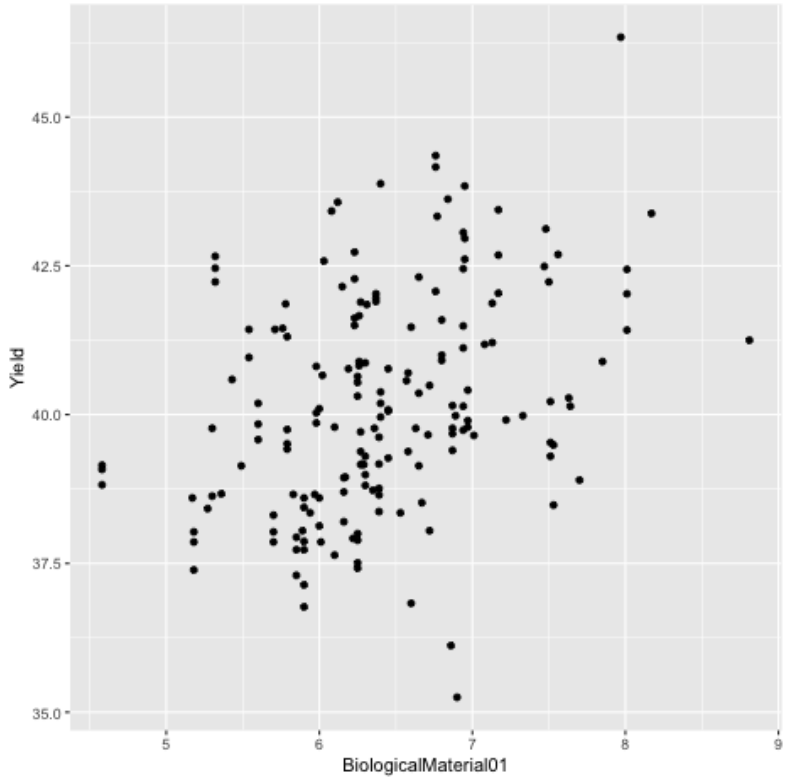
- Since the scatterplot is a bivariate visualization, it requires a definition of 2 axes:
  - `x-axis` and
  - `y-axis`
- In `ggplot` terms, this means that we need to map 2 `aes` parameters (`x` and `y` respectively)
- Let's plot `BiologicalMaterial01` on `x-axis` and `Yield` on `y-axis`

Making a scatterplot with `geom_point`: setup

```
# Let's create a base plot.
ggp2 = ggplot(CMP_subset,
              aes(x = BiologicalMaterial01,
                  y = Yield))
ggp2
```

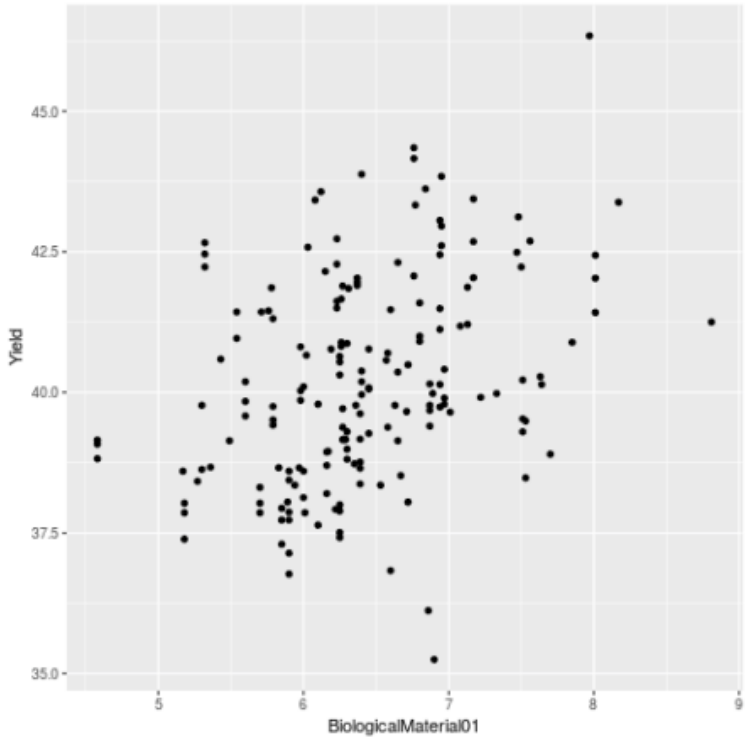



```
# Let's add a layer.
# Each layer is called a `geom`, we need to use
# `geom_point` to add a scatterplot layer.
ggp2 = ggp2 + geom_point()
ggp2
```

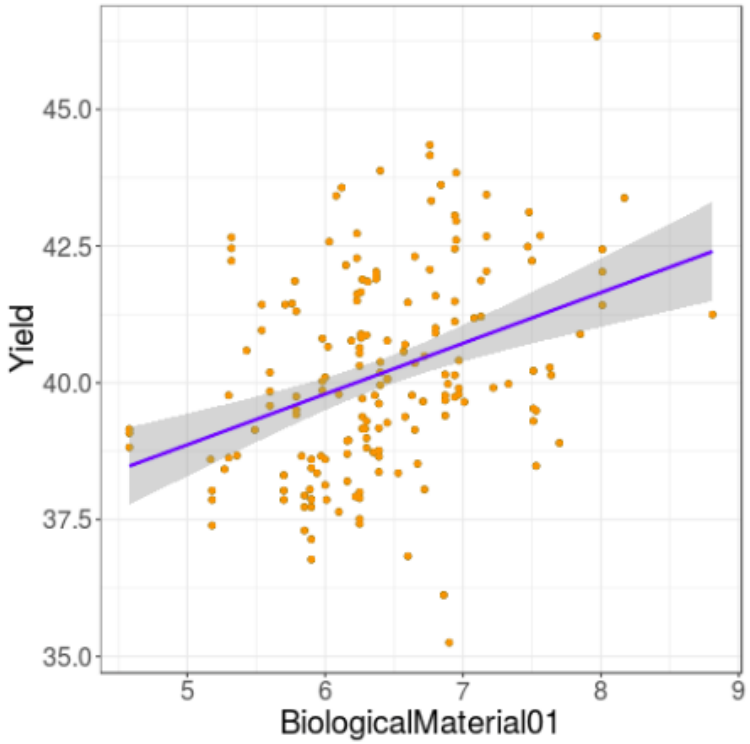


Building a scatterplot with fitted line

What we have



What we need

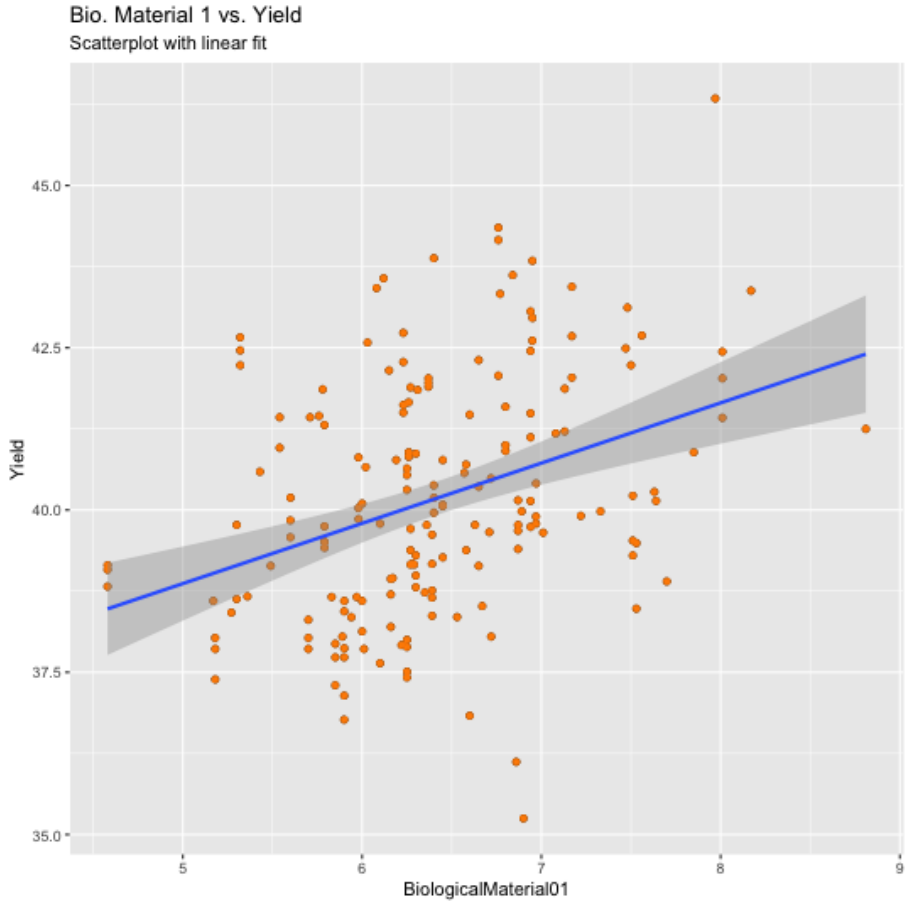


Making a scatterplot with `geom_point`

Adjust

```
ggp2 = ggp2 +

  # Adjust the color of the points.
  geom_point(color = "darkorange") +

  # Add linear regression line (`lm`).
  geom_smooth(method = lm) +

  # Add a title and a subtitle.
  labs(title = "Bio. Material 1 vs. Yield",
       subtitle = "Scatterplot with linear fit")
```

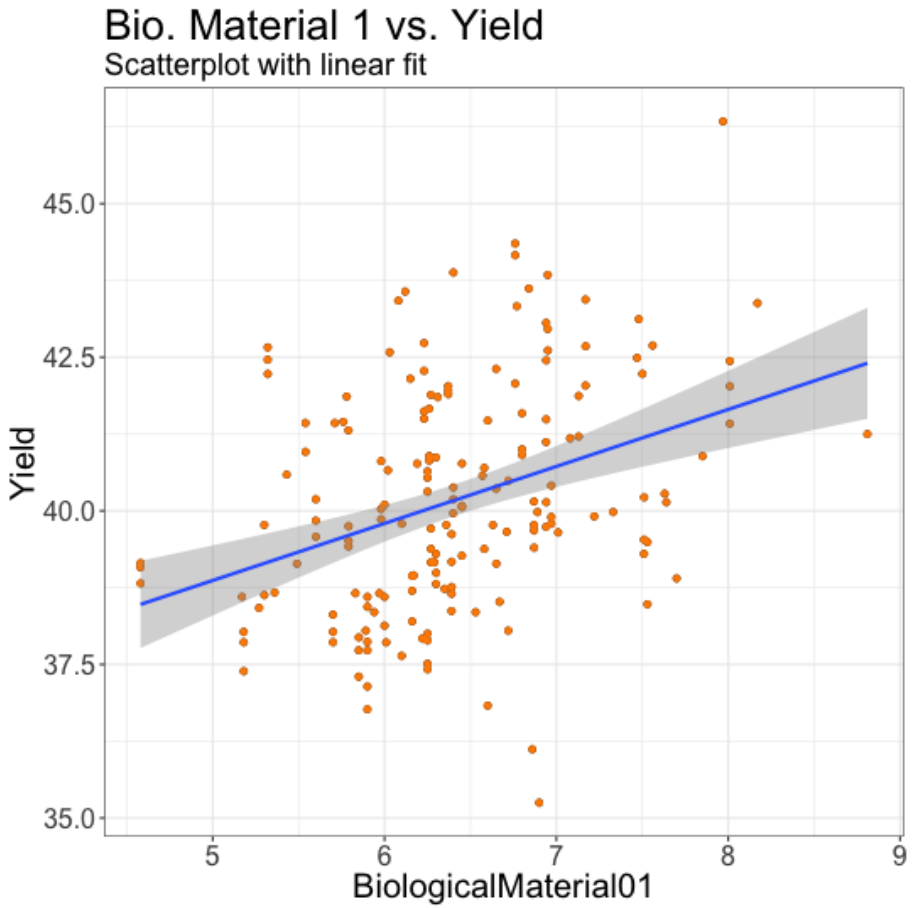
```
# View the plot.
ggp2
```



Making a scatterplot with `geom_point`

Polish

```
ggp2 +

  # Add black & white theme, adjust labels.
  theme_bw() +

  # Customize elements of the theme.
  theme(axis.title = element_text(size = 20),
        axis.text = element_text(size = 16),
        plot.title = element_text(size = 25),
        plot.subtitle = element_text(size = 18))
```



Saving a theme to a variable

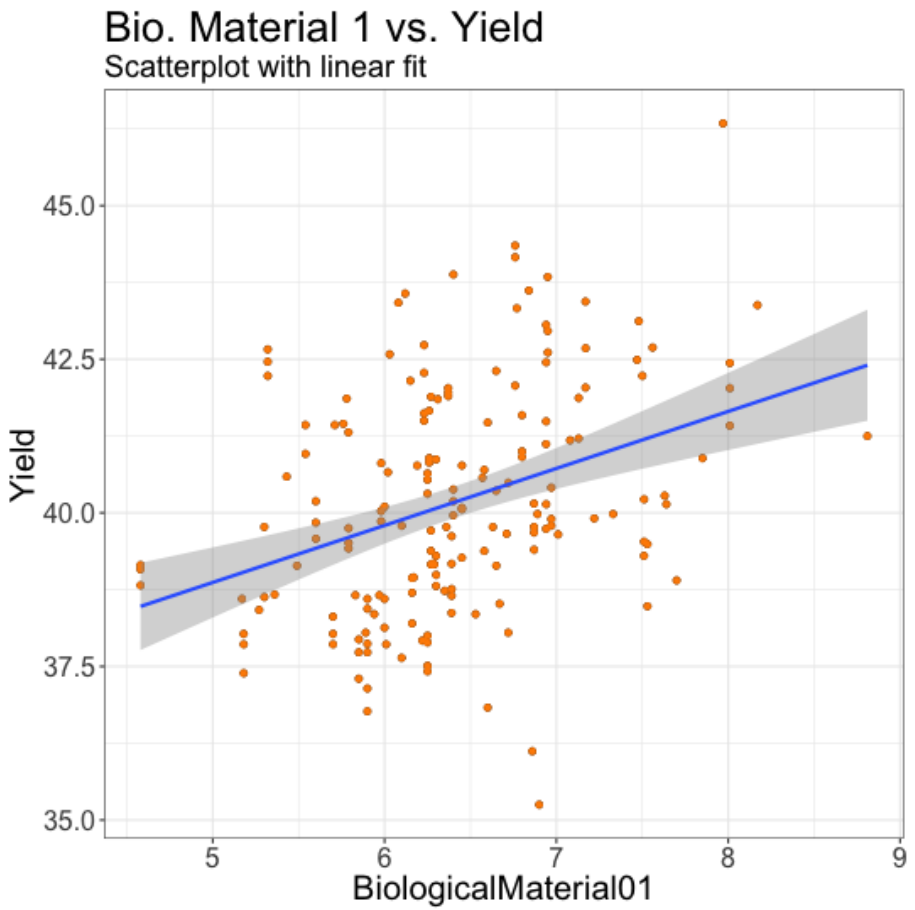
- Notice we've used the following lines of code for both charts:

```
theme_bw() +
  theme(axis.title = element_text(size = 20),
        axis.text = element_text(size = 16),
        plot.title = element_text(size = 25),
        plot.subtitle = element_text(size = 18))
```

- It would be a lot easier to just save these theme adjustments to a variable and add a variable to our `ggplot`

```
my_ggtheme = theme_bw() +
  theme(axis.title = element_text(size = 20),
        axis.text = element_text(size = 16),
        plot.title = element_text(size = 25),
        plot.subtitle = element_text(size = 18))
```

```
# Add saved theme and re-save the plot.
ggp2 = ggp2 + my_ggtheme
ggp2
```



Knowledge check 4

Exercise 4

Module completion checklist

Objective	Complete
Define exploratory data analysis (EDA) and its cycle	✔
Define and differentiate between static and interactive visualizations	✔
Choose when to use univariate plots to illustrate patterns in data	✔
Construct univariate plots using base R	✔
Choose when to use bivariate plots to illustrate patterns in data	✔
Construct bivariate plot using base R	✔
Choose when to use multivariate plots to illustrate patterns in data	✔
Construct multivariate plot (scatterplot matrix) using base R	✔
Measure and observe correlation using corrplot package	✔
Formulate the process of using ggplot2 to build complex plots	✔
Build univariate plots using various geom layers with ggplot2	✔
Build bivariate plots using various geom layers with ggplot2	✔

Workshop!

- Workshops are to be completed in the afternoon either with a dataset for a capstone project or with another dataset of your choosing
- Make sure to annotate and comment your code so that it is easy for others to understand what you are doing
- This is an exploratory exercise to get you comfortable with the content we discussed today
- Today, you will:
  - create univariate plots for the important variables in your dataset
  - analyze the distributions by reading the plots
  - create bivariate plots for pair of variables to find the relationship between the variables
  - create multivariate plots to understand the dataset as a whole

Congratulations on completing this module!