**Day\_2\_session\_02**

**Module IV: Visualization and plotting**

**Plotting with ggplot2**

The aim of this session is to quickly grasp how to create effective and visually appealing graphics using ggplot2. We will cover the fundamentals of ggplot() along with some practical techniques to construct essential plots efficiently. ggplot() is a powerful tool for generating intricate plots with minimal code, thanks to its foundation in the comprehensive grammar of graphics.

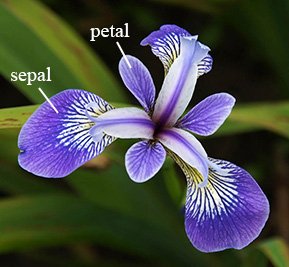
Every ggplot2 plot consists of three essential elements:

**Data:** This should be in the form of a dataframe.

**Aesthetic mappings:** These establish the connections between variables in the data and visual attributes.

**At least one layer:** Layers define how each observation should be displayed and are typically constructed using a geom function.

In this tutorial, we will employ the iris dataset. To better understand the data, it's helpful to have some knowledge about the subject matter. The provided image illustrates the distinction between the petal and sepal of an iris flower:



##Let’s start from a very early stage

ggplot()

ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width))

ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width)) + geom\_point()

ggplot(iris, aes(x=Petal.Length, y=Petal.Width)) + geom\_point()

Let’s say we wish to swap the axes. We can do this using coord\_flip()

plen <- ggplot(data=iris, aes(x=Petal.Length, y=Petal.Width))

plen +

geom\_point() +

coord\_flip()

A good plot should be self-contained. So, add the title and give the names to axes

plen +

geom\_point() +

ggtitle("Iris Petal Size Analysis\nData Source: Anderson (1935)") +

labs(x="Petal Length (cm)", y="Petal Width (cm)")

Let’s now customize the above with theme():

ptcolor <- 'grey20' *# plot text color*

plen +

geom\_point() +

ggtitle("Iris Petal Size Analysis\nData Source: Anderson (1935)") +

labs(x="Petal Length (cm)", y="Petal Width (cm)") +

theme(

plot.title=element\_text(size=14, lineheight=0.8, color=ptcolor, hjust=0.5),

axis.title.x=element\_text(color=ptcolor),

axis.title.y=element\_text(color=ptcolor))

* Differentiate by colour: aes(colour=Species)
* Increase the size of points: size=2
* Multiple customizations: aes(colour=Species), shape=15, size=1.5

ggplot(data=iris, aes(x=Sepal.Length, y=Sepal.Width)) +

geom\_point(aes(colour=Species), shape=15, size=1.5) +

ggtitle("Iris Sepal Size Analysis\nData Source: Anderson (1935)") +

labs(x="Sepal Length (cm)", y="Sepal Width (cm)")

**## Generate a histogram**

ggplot(iris, aes(x=Petal.Length, fill=Species)) +

geom\_histogram(binwidth = 0.1)

## Three separate histograms

ggplot(iris, aes(x = Petal.Length, fill = Species)) +

geom\_histogram(binwidth = 0.75) +

facet\_wrap(~Species) +

scale\_fill\_manual(values = c("setosa" = "red", "versicolor" = "blue", "virginica" = "green")) +

labs(

title = "Histograms of Petal Length by Species",

x = "Petal Length",

y = "Frequency"

)

**###Plot geoms**

* geom\_smooth() fits a smoother to the data and displays the smooth and its standard error.
* geom\_boxplot() produces a box-and-whisker plot to summaries the distribution of a set of points.
* geom\_histogram() and geom\_freqpoly() show the distribution of continuous variables.
* geom\_bar() shows the distribution of categorical variables.
* geom\_path() and geom\_line() draw lines between the data points. A line plot is constrained to produce lines that travel from left to right, while paths can go in any direction. Lines are typically used to explore how things change over time.

### A composite graph by compiling several graphs

**library**(gridExtra)

plen <- ggplot(iris, aes(x=Petal.Length, fill=Species)) +

geom\_histogram(binwidth = 0.1)

pwth <- ggplot(iris, aes(x=Petal.Width, fill=Species)) +

geom\_histogram(binwidth = 0.1)

slen <- ggplot(iris, aes(x=Sepal.Length, fill=Species)) +

geom\_histogram(binwidth = 0.1)

swth <- ggplot(iris, aes(x=Sepal.Width, fill=Species)) +

geom\_histogram(binwidth = 0.1)

grid.arrange(plen, pwth, slen, swth, nrow = 2)

**## Density plots**

As a variation, we can do a density plot instead of a histogram. Note the use of get() within aes() when a variable is used:

common\_subplot <- **function**(data, field) {

ggplot(data, aes(x=get(field), fill=Species)) +

geom\_density(alpha=0.4) +

labs(x=field, y="Density")

}

plen <- common\_subplot(iris, "Petal.Length")

pwth <- common\_subplot(iris, "Petal.Width")

slen <- common\_subplot(iris, "Sepal.Length")

swth <- common\_subplot(iris, "Sepal.Width")

grid.arrange(plen, pwth, slen, swth, nrow = 2)

**## A practice problem using mpg dataset**

[library](https://rdrr.io/r/base/library.html)([ggplot2](https://ggplot2.tidyverse.org/))

data(mpg)

View(mpg)

The variables are mostly self-explanatory:

* cty and hwy record miles per gallon (mpg) for city and highway driving.
* displ is the engine displacement in litres.
* drv is the drivetrain: front wheel (f), rear wheel (r) or four wheel (4).
* model is the model of car. There are 38 models, selected because they had a new edition every year between 1999 and 2008.
* class is a categorical variable describing the “type” of car: two-seater, SUV, compact, etc.

[ggplot](https://ggplot2.tidyverse.org/reference/ggplot.html)(mpg, [aes](https://ggplot2.tidyverse.org/reference/aes.html)(x = displ, y = hwy)) +

[geom\_point](https://ggplot2.tidyverse.org/reference/geom_point.html)()

we could map the class variable to color:

[ggplot](https://ggplot2.tidyverse.org/reference/ggplot.html)(mpg, [aes](https://ggplot2.tidyverse.org/reference/aes.html)(displ, hwy, colour = class)) +

[geom\_point](https://ggplot2.tidyverse.org/reference/geom_point.html)()

If we want to set an aesthetic to a fixed value, without scaling it, do so in the individual layer outside of [aes()](https://ggplot2.tidyverse.org/reference/aes.html). Compare the following two plots:

[ggplot](https://ggplot2.tidyverse.org/reference/ggplot.html)(mpg, [aes](https://ggplot2.tidyverse.org/reference/aes.html)(displ, hwy)) + [geom\_point](https://ggplot2.tidyverse.org/reference/geom_point.html)([aes](https://ggplot2.tidyverse.org/reference/aes.html)(colour = "blue"))

[ggplot](https://ggplot2.tidyverse.org/reference/ggplot.html)(mpg, [aes](https://ggplot2.tidyverse.org/reference/aes.html)(displ, hwy)) + [geom\_point](https://ggplot2.tidyverse.org/reference/geom_point.html)(colour = "blue")

ggplot(mpg, aes(displ, hwy)) + geom\_point(colour = "blue")

ggplot(mpg, aes(displ, hwy)) +

geom\_point(colour = "blue") +

facet\_wrap(~class)

[ggplot](https://ggplot2.tidyverse.org/reference/ggplot.html)(mpg, [aes](https://ggplot2.tidyverse.org/reference/aes.html)(displ, hwy)) +

[geom\_point](https://ggplot2.tidyverse.org/reference/geom_point.html)() +

[geom\_smooth](https://ggplot2.tidyverse.org/reference/geom_smooth.html)()

[ggplot](https://ggplot2.tidyverse.org/reference/ggplot.html)(mpg, [aes](https://ggplot2.tidyverse.org/reference/aes.html)(displ, hwy)) +

[geom\_point](https://ggplot2.tidyverse.org/reference/geom_point.html)() +

[geom\_smooth](https://ggplot2.tidyverse.org/reference/geom_smooth.html)(method = "lm")

[library](https://rdrr.io/r/base/library.html)(mgcv)

[ggplot](https://ggplot2.tidyverse.org/reference/ggplot.html)(mpg, [aes](https://ggplot2.tidyverse.org/reference/aes.html)(displ, hwy)) +

[geom\_point](https://ggplot2.tidyverse.org/reference/geom_point.html)() +

[geom\_smooth](https://ggplot2.tidyverse.org/reference/geom_smooth.html)(method = "gam", formula = y ~ [s](https://rdrr.io/pkg/mgcv/man/s.html)(x))

To compare the distributions of different subgroups, you can map a categorical variable to either fill (for geom\_histogram()) or colour (for geom\_freqpoly()).

[ggplot](https://ggplot2.tidyverse.org/reference/ggplot.html)(mpg, [aes](https://ggplot2.tidyverse.org/reference/aes.html)(displ, colour = drv)) +

[geom\_freqpoly](https://ggplot2.tidyverse.org/reference/geom_histogram.html)(binwidth = 0.5)

[ggplot](https://ggplot2.tidyverse.org/reference/ggplot.html)(mpg, [aes](https://ggplot2.tidyverse.org/reference/aes.html)(displ, fill = drv)) +

[geom\_histogram](https://ggplot2.tidyverse.org/reference/geom_histogram.html)(binwidth = 0.5) +

[facet\_wrap](https://ggplot2.tidyverse.org/reference/facet_wrap.html)(~drv, ncol = 1)

**## Dynamic and interactive visualization**

library("highcharter")

data(diamonds, mpg, package = "ggplot2")

View(diamonds)

View(mpg)

hchart(mpg, "scatter", hcaes(x = displ, y = hwy, group = class))

?hchart

highchart() %>%

hc\_chart(type = "column") %>%

hc\_title(text = "A highcharter chart") %>%

hc\_xAxis(categories = 2012:2016) %>%

hc\_add\_series(data = c(3900, 4200, 5700, 8500, 11900),

name = "Downloads")

hchart(diamonds$cut, colorByPoint = TRUE, name = "Cut")

hchart(diamonds$price, color = "#B71C1C", name = "Price") %>%

hc\_title(text = "You can zoom me")

library("quantmod")

x <- getSymbols("GOOG", auto.assign = FALSE)

y <- getSymbols("AMZN", auto.assign = FALSE)

highchart(type = "stock") %>%

hc\_add\_series(x) %>%

hc\_add\_series(y, type = "ohlc")

data(unemployment)

View(unemployment)

hcmap("countries/us/us-all-all", data = unemployment,

name = "Unemployment", value = "value", joinBy = c("hc-key", "code"),

borderColor = "transparent") %>%

hc\_colorAxis(dataClasses = color\_classes(c(seq(0, 10, by = 2), 50))) %>%

hc\_legend(layout = "vertical", align = "right",

floating = TRUE, valueDecimals = 0, valueSuffix = "%")

Ref:

* Hadley Wickham’s Advanced R book
* Roger Peng’s R Programming for Data Science book
* DataCamp’s Intermediate R course
* Coursera’s R Programming course
* https://ggplot2-book.org/getting-started.html
* Data Carpentry (http://datacarpentry.org/), data camp, data quest, Kaggle
* Harvard Chan Bioinformatics Core (HBC) under the open access terms of the Creative Commons Attribution license (CC BY 4.0),
* The Book of R: A First Course in Programming and Statistics by Tilman M. Davies
* UC Business Analytics R Programming Guide and R\_bootcamp