**ggplot2 tutorial**

**References and useful information**

Examples based on <https://r4ds.had.co.nz/data-visualisation.html>

List of functions and function reference [https://ggplot2.tidyverse.org/reference/#section-plot-basics](https://ggplot2.tidyverse.org/reference/" \l "section-plot-basics)

Useful for understanding geometry layers https://rpubs.com/hadley/ggplot2-layers

Further exercises and advanced functionality <https://r4ds.had.co.nz/graphics-for-communication.html>

There are entire books on ggplot2

Complex but flexible and powerful plotting library

Based on ‘the grammar of graphics

**ggplot components**

ggplot object

geom layers

aes mapping

scales

faceting specifications

coordinate systems

ggplot() object

creates a coordinate system to which layers can be added

arg 1: data = ‘dataset’

geom layers

geom\_point() adds a layer of points creating a scatterplot

aes() mapping

the x and y args of the aes() function are where the variables to be mapped onto coordinate system are given. ggplot looks in the ggplot() object data argument for these variables

**Basic generic use template**

ggplot(data = ‘dataset’) +

‘geom\_function’(mapping = aes(‘mappings’))

**mpg scatterplot example**

ggplot(data=mpg) assign data, blank plot

ggplot(data=mpg) +

geom\_point(mapping=aes(x=displ,y=hwy))

basic scatter plot using geom\_point() layer and mapping of variables within mpg dataframe

Aesthetic properties include size, shape and colour of points

ggplot(data=mpg) +

geom\_point(mapping=aes(x=displ,y=hwy, color=class))

the points are coloured by passing a variable in the dataframe to the the aes argument color

ggplot(data=mpg) +

geom\_point(mapping=aes(x=displ,y=hwy, size=class)) size (continuous variable recommended)

ggplot(data=mpg) +

geom\_point(mapping=aes(x=displ,y=hwy, alpha=class)) transparency (continuous variable recommended)

ggplot(data=mpg) +

geom\_point(mapping=aes(x=displ,y=hwy, shape=class)) shape (only 6 available automatically)

**Manual aesthetics**

manual point aesthetics need to be set outside of the aes mapping as they do not derive from the dataframe

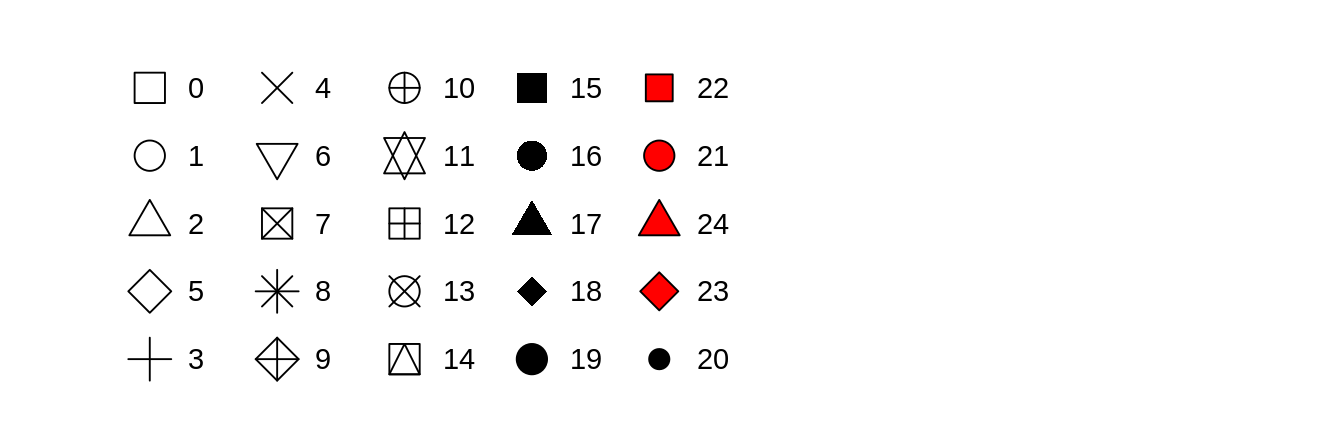
ggplot(data = mpg) +

geom\_point(mapping = aes(x = displ, y = hwy, color = "blue"))

color: name of color as a string

size: size in mm

shape: one of the following numbers below



**Facet wrap**

ggplot(data = mpg) +

geom\_point(mapping = aes(x = displ, y = hwy)) +

facet\_wrap(~ class, nrow = 2)

separate into individual plots by class and arrange in 2 rows

ggplot(data = mpg) +

geom\_point(mapping = aes(x = displ, y = hwy)) +

facet\_grid(drv ~ cyl)

separate into individual plots by drive type and number of cylinders arranged as a grid

**Multiple geometry layers**

ggplot(data = mpg) +

geom\_smooth(mapping = aes(x = displ, y = hwy))

ggplot(data = mpg) +

geom\_smooth(mapping = aes(x = displ, y = hwy, linetype = drv))

ggplot(data = mpg) +

geom\_point(mapping=aes(x=displ,y=hwy, color=drv)) +

geom\_smooth(mapping = aes(x = displ, y = hwy, linetype = drv))

**Global mapping**

ggplot(data = mpg, mapping = aes(x = displ, y = hwy)) +

geom\_point(mapping=aes(color=drv)) +

geom\_smooth(mapping=aes(linetype=drv))

**Bar plot**

ggplot(data=mpg) +

geom\_bar(mapping=aes(x=class))

bar charts, histograms, and frequency polygons bin your data and then plot bin counts, the number of points that fall in each bin.

smoothers fit a model to your data and then plot predictions from the model.

boxplots compute a robust summary of the distribution and then display a specially formatted box.

**Stat transformations**

ggplot(data=mpg) +

stat\_count(mapping=aes(x=class))

**Coordinates**

ggplot(data = mpg, mapping = aes(x = class, y = hwy)) +

geom\_boxplot()

ggplot(data = mpg, mapping = aes(x = class, y = hwy)) +

geom\_boxplot() +

coord\_flip()

bar <- ggplot(data = mpg) +

geom\_bar(

mapping = aes(x = class, fill = class),

show.legend = FALSE,

width = 1

) +

theme(aspect.ratio = 1) +

labs(x = NULL, y = NULL)

bar + coord\_flip()

bar + coord\_polar()

**The structure of graphical grammar**

ggplot(data = <DATA>) +

<GEOM\_FUNCTION>(

mapping = aes(<MAPPINGS>),

stat = <STAT>,

position = <POSITION>

) +

<COORDINATE\_FUNCTION> +

<FACET\_FUNCTION>