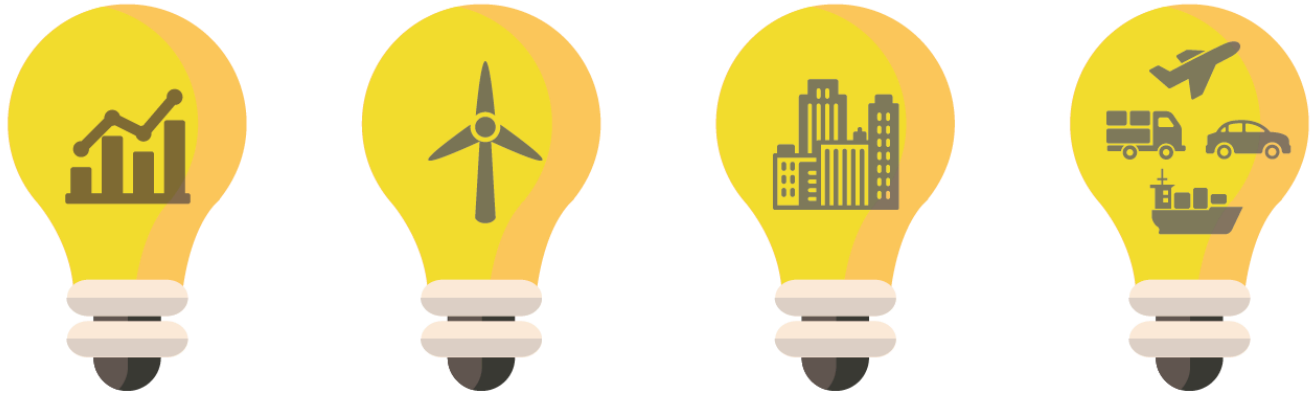
BENV0091 Lecture 2: Programming & Visualisation

Patrick de Mars



Lecture Overview

- Part 1: Programming
  - Functions
  - If-Else
  - Iteration
- Part 2: Introduction to Visualisation with ggplot2

Part 1: Programming

Setting up

- Task: open RStudio and create a new project called lecture2, making it a subdirectory of BENV0091 from last week
- Open a new R Notebook or Markdown file and save it as programming.Rmd in your lecture2 directory

Functions

- We have been introduced to several functions such as print(), mutate(), mean() etc.
- Functions can be useful to avoid **repetition**
- Functions also improve **readability**
- We can write our own functions in R (using the syntax on the right)
- Task: write a function addition(x, y) which **returns** the sum of two arguments
  - *Test your function with different inputs, making sure it gives the correct answers*

```
func_name <- function(arg1, arg2,...){
  your_code()
}
```

*If you want your function to output something, use **return(output)***

If you do not specify return(…), the function returns NULL

Functions: Exercises

```
func_name <- function(arg1, arg2,...){
  your_code()
}
```

1. Write a function to calculate the interquartile range of a **vector**
2. Write the following functions for the mpg dataframe:
    1. Calculate the average hwy MPG for a specified manufacturer
    2. Return the highest cty MPG for a specified class and drv
    3. Calculate the correlation between cty MPG and displ for a specified class

Test out your functions with different inputs: make sure they give sensible answers!

Create a vector with c(x, y,…)

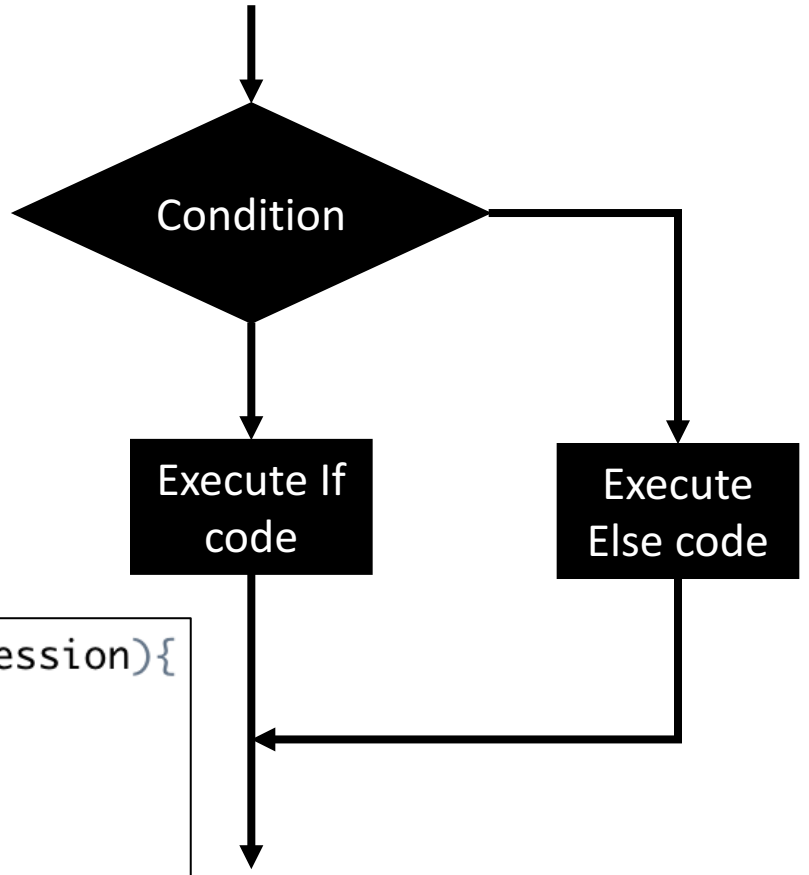
Use quantile(vector, x) to find the x (as decimal) quantile of a numeric vector

The mpg dataframe is built into ggplot: it is already assigned to an object named mpg

Use cor(x, y) to calculate the correlation between vectors x and y

If-Else

- The If-Else statement is an essential building block of programming
- It relies on logical expressions that evaluate to TRUE or FALSE (**logical variables**)
- Task: write a function that prints “Even” or “Odd” depending if the input an even/odd number
- Task: modify the function to print “Not an integer” if the input is not an integer




```
if (logical_expression){
  if_code()
} else{
  else_code()
}
```

```
if (logical_expression){
  do_something()
}
```

x %% y gives the remainder of x/y

Exercises

1. Write a function that checks if a letter is a vowel (return TRUE or FALSE)
2. Using a nested if-else statement, write a function that prints:
   - ”Big and even!” if input is an even number that is greater than 10
   - “Big and odd!” if input is an odd number that is greater than 10
   - ”Small :(” otherwise
3. Write a function that finds the roots of a quadratic equation given coefficients a, b, c
4. Write a function to change a word to lower case and remove vowels

Nested if-else statements: if-else within if-else!

Quadratic formula

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Tip: tolower()

Tip: str_remove_all()

Vectorised If-Else

- It can be useful to use If-Else to transform data
- The ifelse() (base R) and if_else() (dplyr) functions are designed to make it easy to apply If-Else statements to vectors (e.g. in data frames)
- Task: add a `engine_category` column to mpg that is “large” for cars with at least a 2L engine displacement (`displ`), and small for all other cars
  - Use count() to determine the number of cars with large and small engines

if_else(condition, if_result, else_result)

Iteration: For Loops

```
for (value in sequence){
  do_something()
}
```

- Often you will want to perform the same operation on multiple inputs:
  - Reading multiple data frames
  - Calculating statistics for multiple columns
- A for loop is a handy way to iterate over a **vector** (i.e. a sequence of values)
- Task: write a for loop that prints the numbers 1 to 100
  - Use an if statement to print only multiples of 7

Create a sequence of integers from x to y with x:y

For Loop Exercises

1. Write a for loop that reads each of the CSV files in the beis_headcount directory
2. For each year in the beis_headcount directory find out the following:
   - Total headcount
   - Headcount of the Committee for Climate Change
   - Headcount of the UK Space Agency
   - Department with the largest headcount
3. Write a for loop to read all of the CSV files and combine them into a single data frame
   - Try adding a `year` column to each data frame before combining

list.files(directory) creates a vector all files in directory

file.path(x, y) joins strings x and y into a single path

str_extract(x, "[0-9]+") extracts all numbers from x

Tip: bind_rows()

Iteration: Map

- A less verbose and more elegant alternative to the for loop is to use a map function from the **purrr** package
- Map applies a function to each element of a vector or list
- Task: use map() to get the square root of the vector `c(1, 4, 9, 16, 25)`
- As data frames are really just a **list of equal-length vectors**, map() can be used to apply a function to every column
- Task: use map() and mode() to get the type of each column in the `mpg` data frame

map(data, function) applies function to all elements in data

While map() returns a list, map_dbl() and map_chr() return vectors of types double and character

Note: you can always use for loops, but you may find map() more elegant

Part 2: Visualisation

Data: Canadian Wind Turbines

- We will be using data on Canadian wind turbines (a Tidy Tuesday dataset)
- Download from Moodle and add to your lecture2/data directory
- Open a new R Notebook and save it as visualisation.Rmd
- Read the CSV file and assign it to an object called `turbines`

ggplot2

ggplot2: Elegant Graphics for Data Analysis (book)

- We will be using the ggplot2 (tidyverse) package for visualisation in R
- gg stands for “grammar of graphics”
- ggplot uses layers to iteratively build complex plots
- Task: run the code below to create a scatter plot of turbine capacity against rotor diameter
- Task: try changing the x and y variables to other columns in the turbine data frame


```
ggplot(data = turbines) +
  geom_point(aes(x = rotor_diameter_m, y = turbine_rated_capacity_k_w))
```

Anatomy of a Plot

Use the + symbol!

Specify the data

```
ggplot(data = turbines) +
  geom_point(aes(x = rotor_diameter_m, y = turbine_rated_capacity_k_w))
```

*Choose a **geom function***

*Specify the **aesthetic mapping***

Adding Aesthetics

Shape, size and color aesthetics

- In addition to x and y aesthetics, we can also specify further aesthetics for points in a scatterplot:
  - colour
  - shape
  - size
  - alpha
- What sort of data (categorical or continuous) is appropriate for each aesthetic?
- Task: assign the size aesthetic to hub height
- Task: color points by province

```
ggplot(data = turbines) +
  geom_point(aes(x = rotor_diameter_m,
                 y = turbine_rated_capacity_k_w),
             color = 'purple',
             alpha = 0.5,
             shape = 2,
             size = 3)
```

You can also set a fixed aesthetic for all points

Bar Charts

- The geom_bar() and geom_col() plots can be used to create bar charts
  - `geom_bar()` counts the number of cases at each x position
  - `geom_col()` leaves the data as it is
- The two geom functions on the right will produce the same plots
- Task: produce a plot of the number of wind turbines in each province

variable	…
A	
A	
A	
B	

geom_bar(aes(x = variable))

variable	count
A	3
B	1

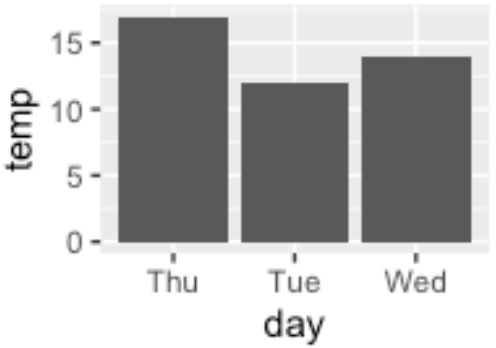
geom_col(aes(x = variable, y = count))

Factors

- When dealing with categorical data, ggplot will implicitly convert **characters** to **factors**
- Factors are used for categorical variables
- Factors have **levels** (categories) whose order you can specify for the purposes of plotting (for instance)

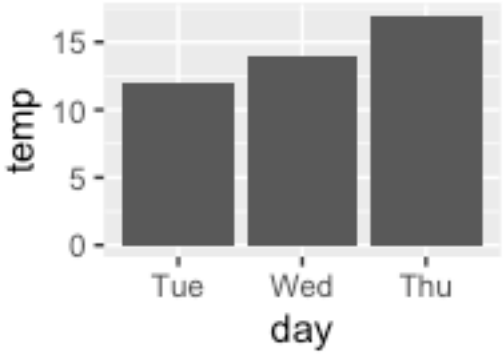
Columns are shown in alphabetical order by default

```
df <- tibble(day = c('Thu', 'Tue', 'Wed'),
             temp = c(17, 12, 14))

df %>%
  ggplot() +
  geom_col(aes(x = day, y = temp))
```



Factors are reordered manually

```
df <- tibble(day = c('Thu', 'Tue', 'Wed'),
             temp = c(17, 12, 14)) %>%
  mutate(day = fct_relevel(day, 'Tue', 'Wed', 'Thu'))

df %>%
  ggplot() +
  geom_col(aes(x = day, y = temp))
```



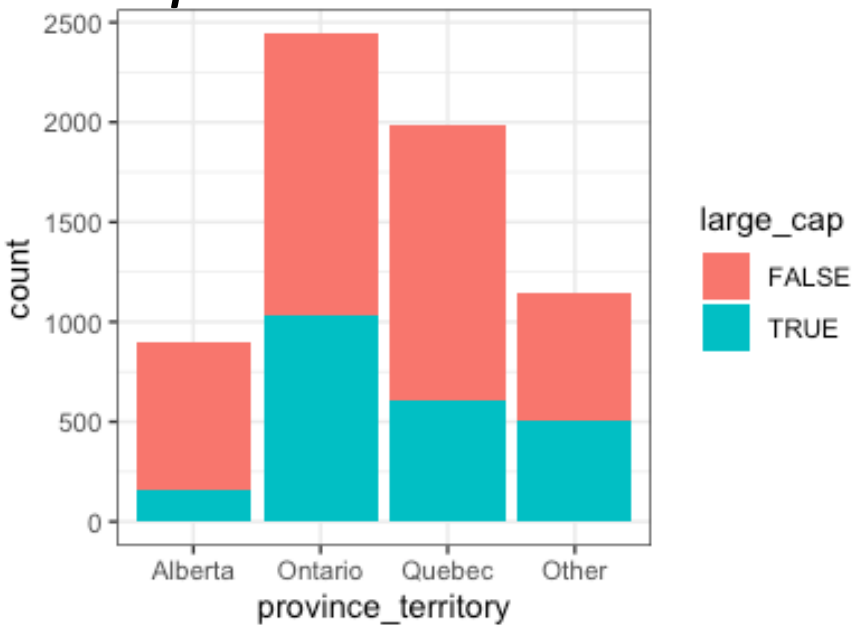
Forcats Functions

- Useful functions from the forcats package include
  - fct_lump(): group smaller categories into an ‘Other’ category
  - fct_infreq(): order a factor by frequency (number of appearances of that category)
  - fct_reorder(): order a factor by another variable
- Type vignette(‘forcats’) for some examples
- Task: use **fct_reorder()** or **fct_infreq()** to order the bars in the wind turbine bar chart (from decreasing to increasing)
- Task: use **fct_lump()** to reduce the number of provinces to 3 plus an Other category; then plot turbine capacity vs. height, coloured by province

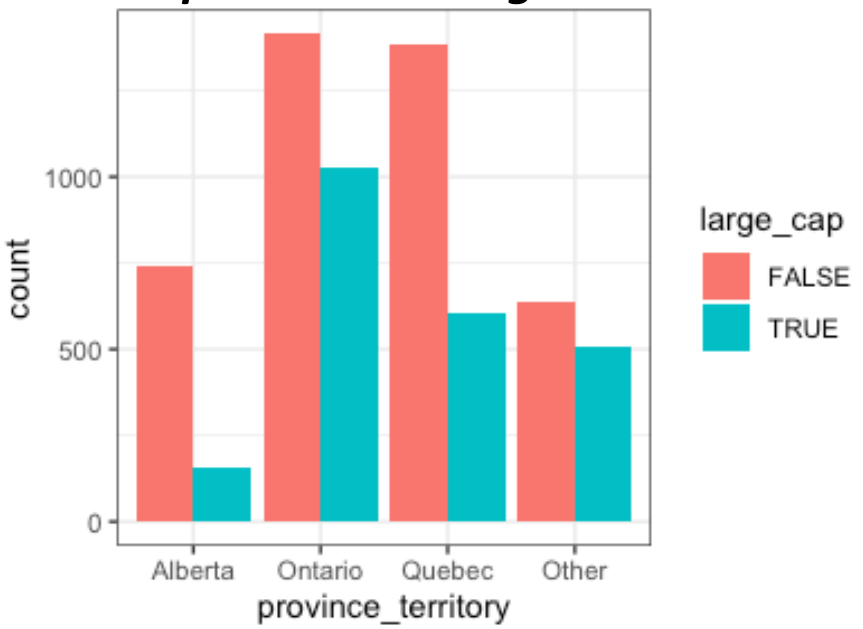
Bar Charts: Fill and Position

- For bar charts, a useful aesthetic is `fill` for determining the fill colour of bars
- When specifying fill, we can either stack bars on top of one another or place them side by side
  - position = ‘stack’
  - position = ‘dodge’
- In the right hand plot a column `large_cap` (capacity > 2 MW) has been added to `turbines` using mutate()

position = ‘stack’

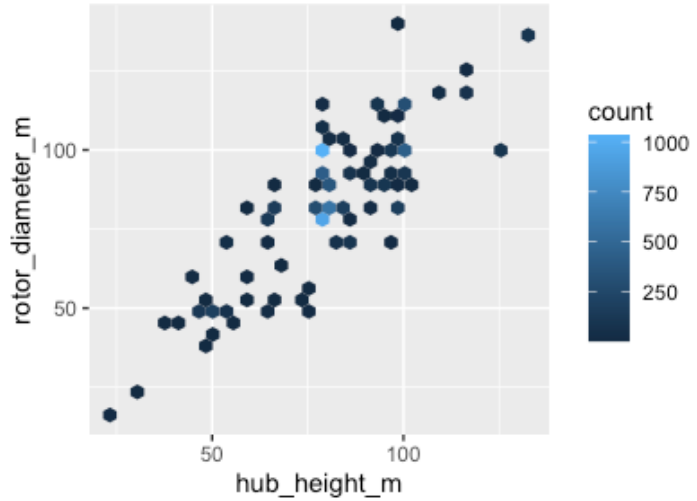


position = ‘dodge’

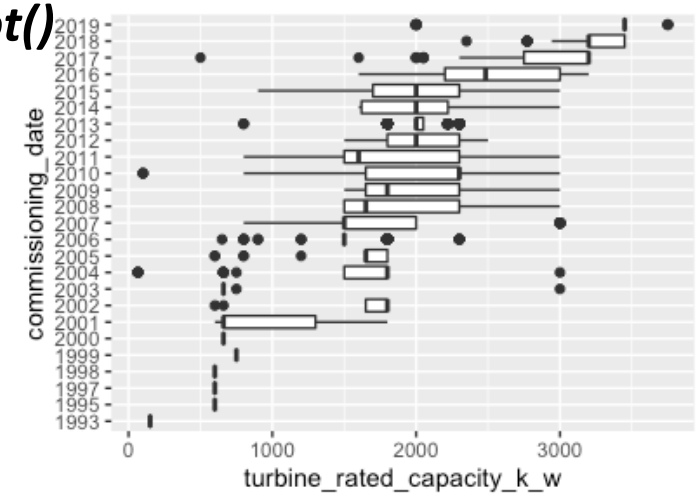


Exercises: Reproduce These Plots!

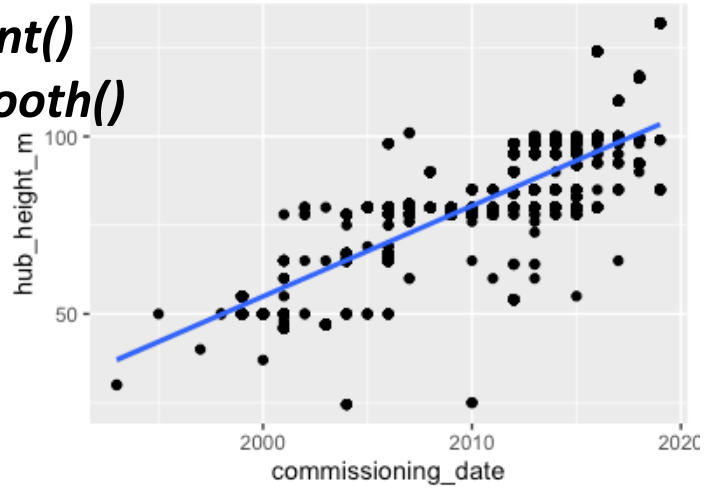
geom_hex()



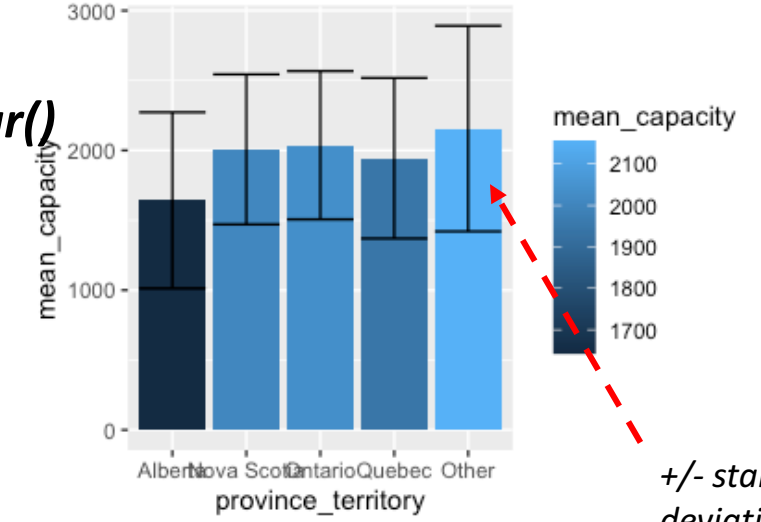
geom_boxplot()



geom_point()
geom_smooth()



geom_col()
geom_errorbar()

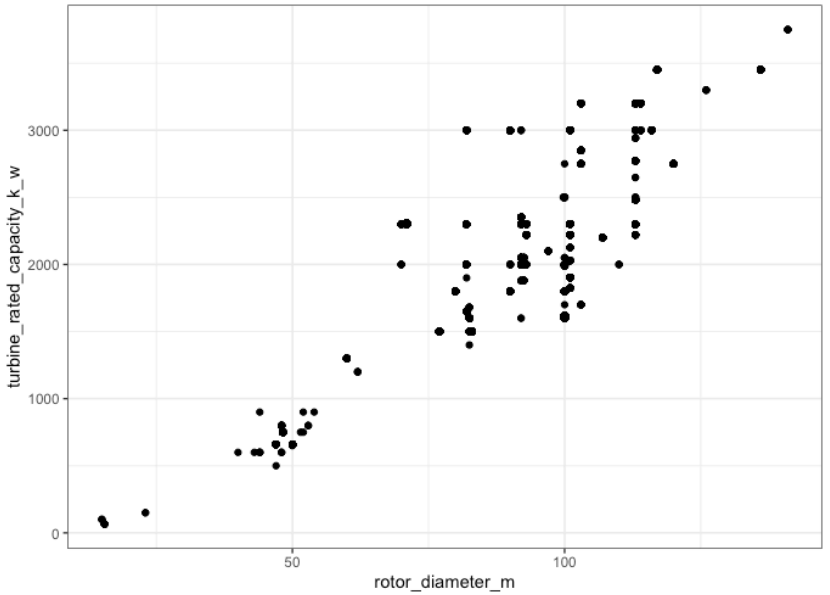


+/- standard deviation!

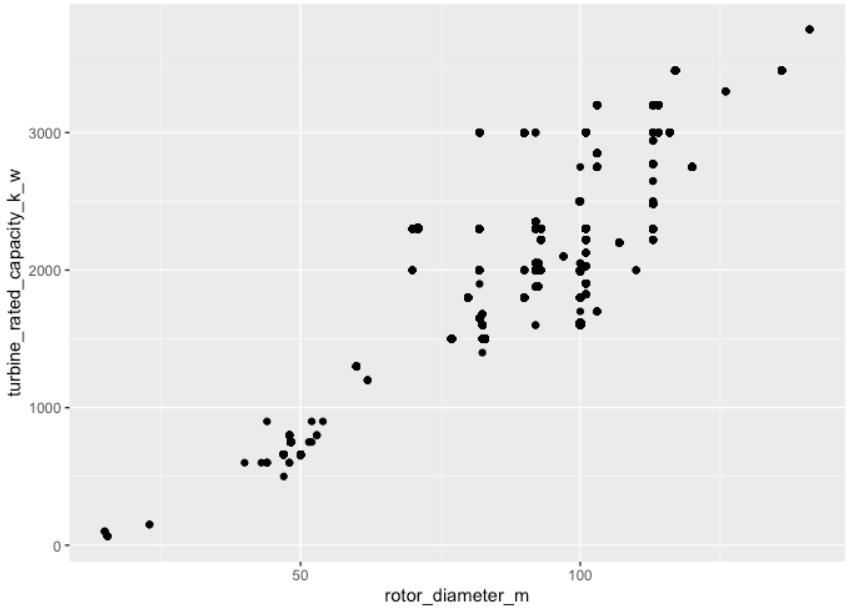
Customising your Plot

- Use labs() to change axis labels and title
- Add a theme to change the colors
- Change the coordinate system:
  - coord_flip()
  - coord_polar()
  - coord_trans()
- Save your plots!

theme_bw()

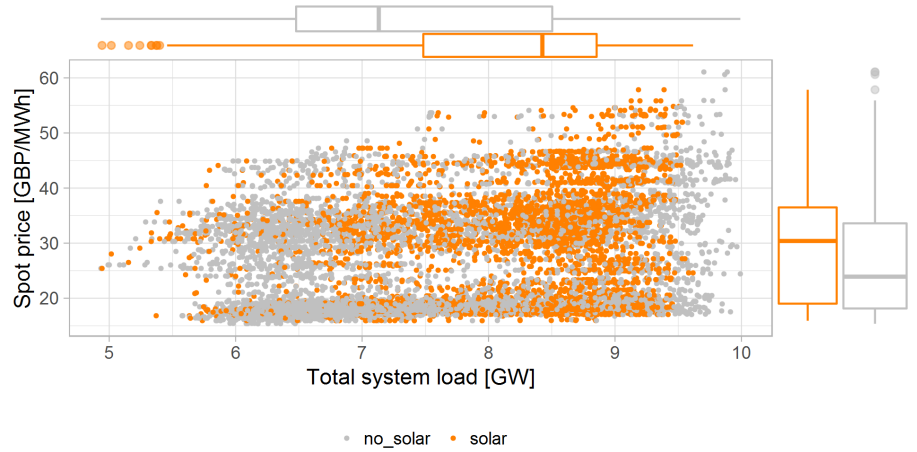


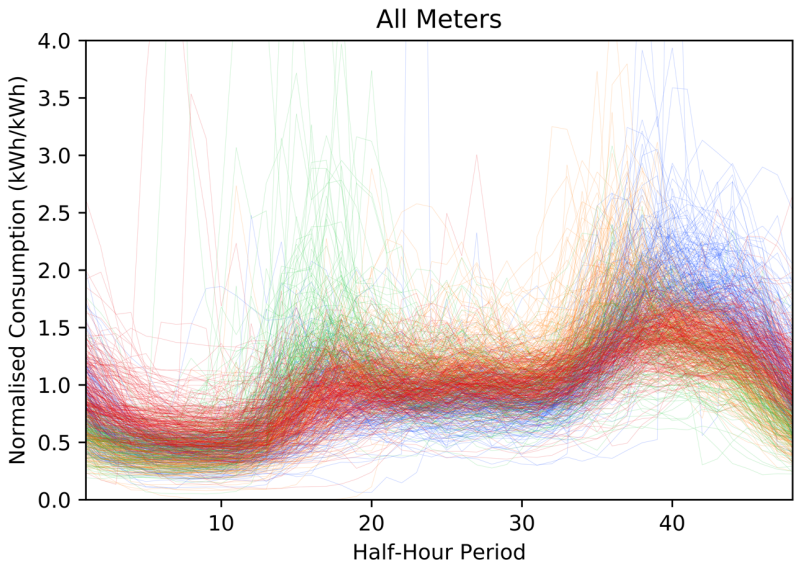
theme_grey()



Visualisation Competition!

- Produce an energy-related plot using a dataset of your choice
- **Single figure**: no gifs/videos
- No maps!
- Visualisations will be judged on:
  - Is it clear?
  - Does it help tell a story?
  - Is it visually appealing?
- £100 prize 😱 😱 😱
- Email your submissions to Patrick with the subject **VISUALISATION** by **10am on Wednesday 27th October**





Mentimeter