ESSA
EXERCISE & SPORTS SCIENCE AUSTRALIA

Making Sports Science Easier Using Data Analysis and Visualisation Programs

Dr Heidi Thornton PhD, AES, ASpS2

Chaired by Simon Price

15 September 2021

About me

👏 I'm an applied sports scientist

I currently work with two 🏈 teams

- Gold Coast Suns Football Club (AFL)
- Newcastle Knights Rugby League Club (NRL)

📌 I live in an pretty amazing place (Newcastle)

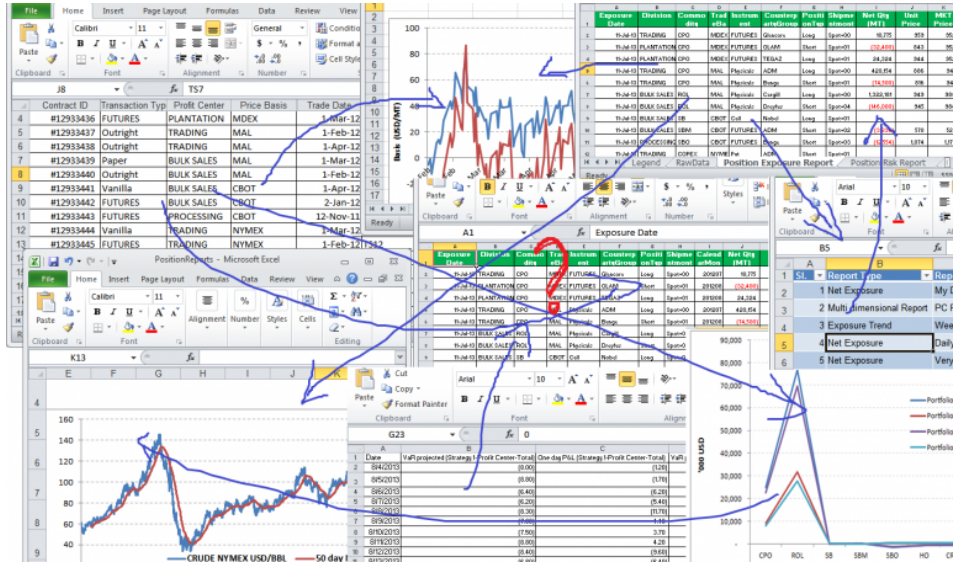
How are you analysing your data?

Why learn a new program?

- Save time
- Produce high quality reports or visualisations
- Staying up to date with the industry
- Personal development
- Analyse complex datasets more easily
- Reproducibility of reports - automate workflows
- SAVE TIME

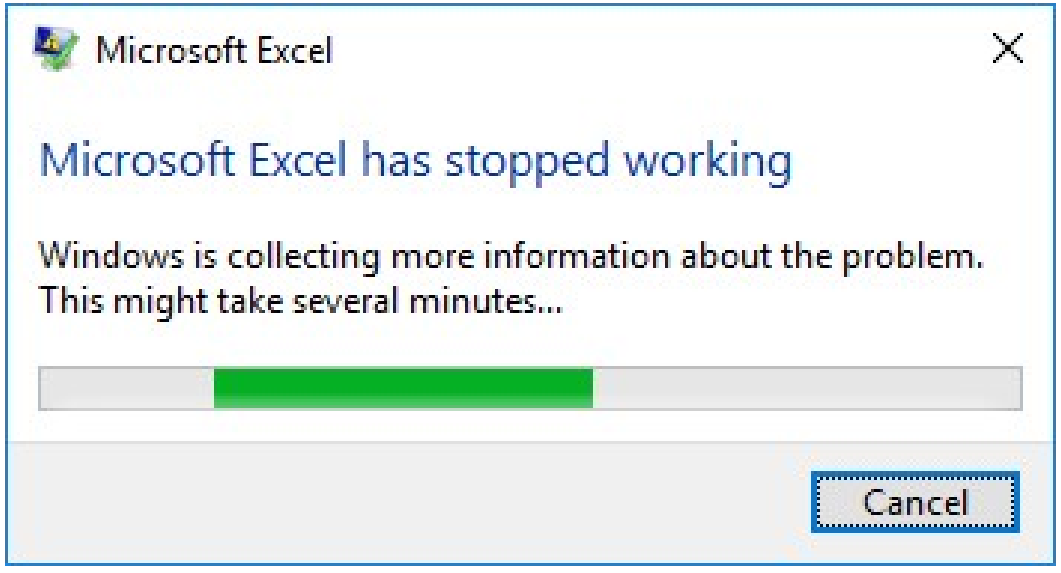
But Excel is working for me

- 👍 Excel is easy to use and has a place in data entry, analysis and visualisation
- 👍 Lots of help readily available
- 👍 Widely used by practitioners in sport
- 👎 However, it has it's limitations



But Excel is working for me

And soon enough....



Credit: Dr Jacquie Tran (ESSA Forum, 2019) and Dr Alice Sweeting (WCSF Workshop, 2019)

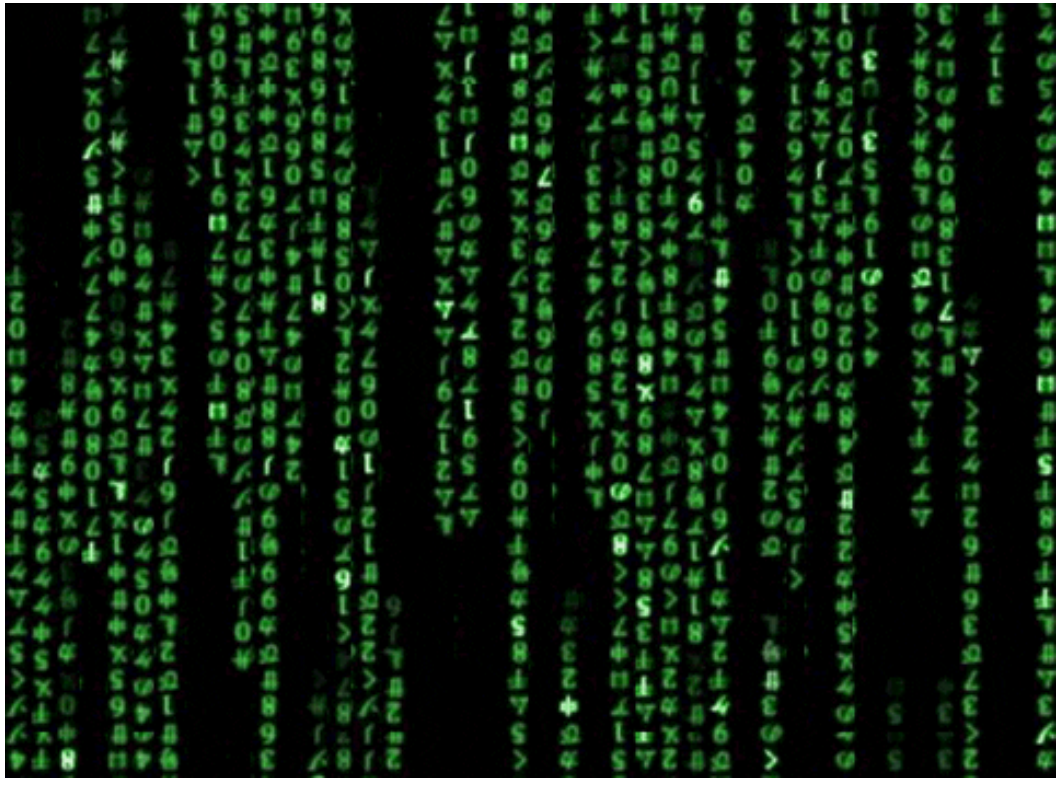
Part 1 - R programming

R programming

? Why use R?

- Reproducibility -> automate workflows
- Simple to really complex analysis
- Its free
- Can create complex dashboards, or simple PDF/HTML outputs
- Fast and efficient analysis
- Handles big data sets really well (i.e., raw GPS files)

What others think R looks like



Credit: Dr Alice Sweeting (WCSF Workshop, 2019)

Where to start with R

- Need to download R first
- I recommend using RStudio which is an interface for R, making it less 'scary' and more user friendly
- Practice using a data set you know and understand - start out with some simple plots

👉 There are tonnes of resources available, even some relevant to sport

Mitch Henderson
@mitchhendo_

🚨 New post & video tutorial 🚨

How sports scientists can use ggplot2 in R to make better visualisations 📊

Great visualisations help communicate your message more clearly. This post shows you an example of my process. #rstats

mitchhenderson.org/2020/04/how-sp...

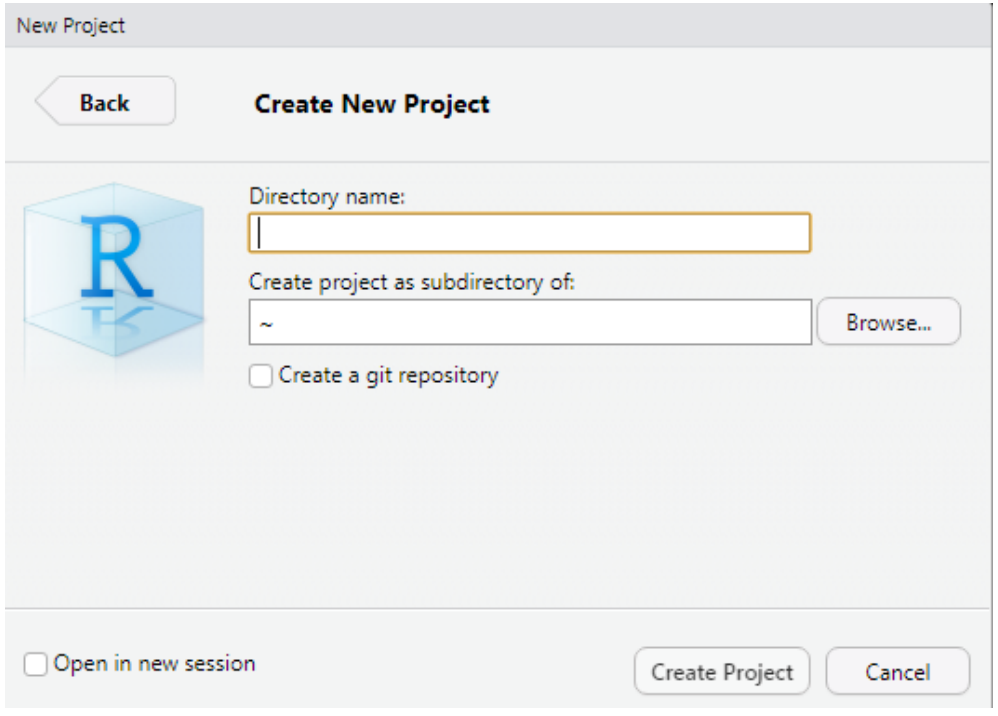
Example 1 - strength data

Create a project

🔍 Projects help to organise files and locations

👉 In RStudio, click File - New Project - New Directory - New Project

👉 Projects allow all files (scripts, figures, output etc) to be stored together



Import strength testing data

Packages provide functions to perform different tasks

```
install.packages("readr")
```

Import .csv strength testing file using **readr** package

```
library(readr)
Strength <- read_csv("Files/Strength.csv")
```

Or import .xlsx file using **readxl** - note the different location to file

```
library(readxl)
Strength_xlsx <- read_excel("C:/Users/Heidi.Thornton/OneDrive - GCFC Limited/Documents/Strength.xlsx")
```

View data

👓 We can view our data set using **print** or **head**

```
print(Strength, n=6)
```

```
head(Strength, n=6)
```

```
## # A tibble: 6 x 8
##   Date       Name      Season Period `Bodyweight (kg~ `3RM Weighted C~ `3RM Bench Pres~
##   <chr>      <chr>      <dbl> <chr>              <dbl>            <dbl>            <dbl>
## 1 1/11/2020 Athlete 1   2021 Start~               85              120               95
## 2 1/11/2020 Athlete 2   2021 Start~               86              125               97.5
## 3 1/11/2020 Athlete 3   2021 Start~               88              122.             100
## 4 3/11/2020 Athlete 3   2021 Start~               89              125              102.
## 5 1/11/2020 Athlete 4   2021 Start~               90              122              110
## 6 1/11/2020 Athlete 5   2021 Start~              100               90              120
## # ... with 1 more variable: 3RM Box Squat <dbl>
```

Summarise data

🖌 Using the **tidyverse** package, we will create some summary statistics from the dataset

```
Strength_summary <- Strength %>%
      group_by(Name) %>%
          filter(Season == 2021) %>%
            summarize(
                      'Max chins (kg)'= max(`3RM Weighted Chins`),
                      'Change chins (kg)' = (max(`3RM Weighted Chins`))-(min(`3RM Weighted Chins`)),
                      'Max bench (kg)'= max(`3RM Bench Press`),
                      'Change bench (kg)' = (max(`3RM Bench Press`))-(min(`3RM Bench Press`)),
                      'Max squat (kg)'= max(`3RM Box Squat`),
                      'Change squat (kg)' = (max(`3RM Box Squat`))-(min(`3RM Box Squat`)))
```

Summarise data

🖌 Using the **tidyverse** package, we will create some summary statistics from the dataset

```
print(Strength_summary, n=10)
```

```
## # A tibble: 8 x 7
##   Name      `Max chins (kg)` `Change chins (k~ `Max bench (kg)` `Change bench (~
##   <chr>                <dbl>              <dbl>            <dbl>             <dbl>
## 1 Athlete 1              125                5               97.5               2.5
## 2 Athlete 2              128.               2.5            108.               10
## 3 Athlete 3              130                7.5            105                 5
## 4 Athlete 4              128.               5.5            115                 5
## 5 Athlete 5               95                5              128.                7.5
## 6 Athlete 6              118.               7.5            108.                2.5
## 7 Athlete 7              110                5              120                 5
## 8 Athlete 8              115                2.5            100                 2.5
## # ... with 2 more variables: Max squat (kg) <dbl>, Change squat (kg) <dbl>
```

Visualise strength data

Create a summary table of strength testing results using **formattable** package

```
library(formattable)

formattable(Strength_summary, align = c('c'), list(
  `Change chins (kg)`= color_tile("white", "#71CA97"),
  `Change bench (kg)`= color_tile ("white","#71CA97"),
  `Change squat (kg)`= color_tile("white","#71CA97")))
```

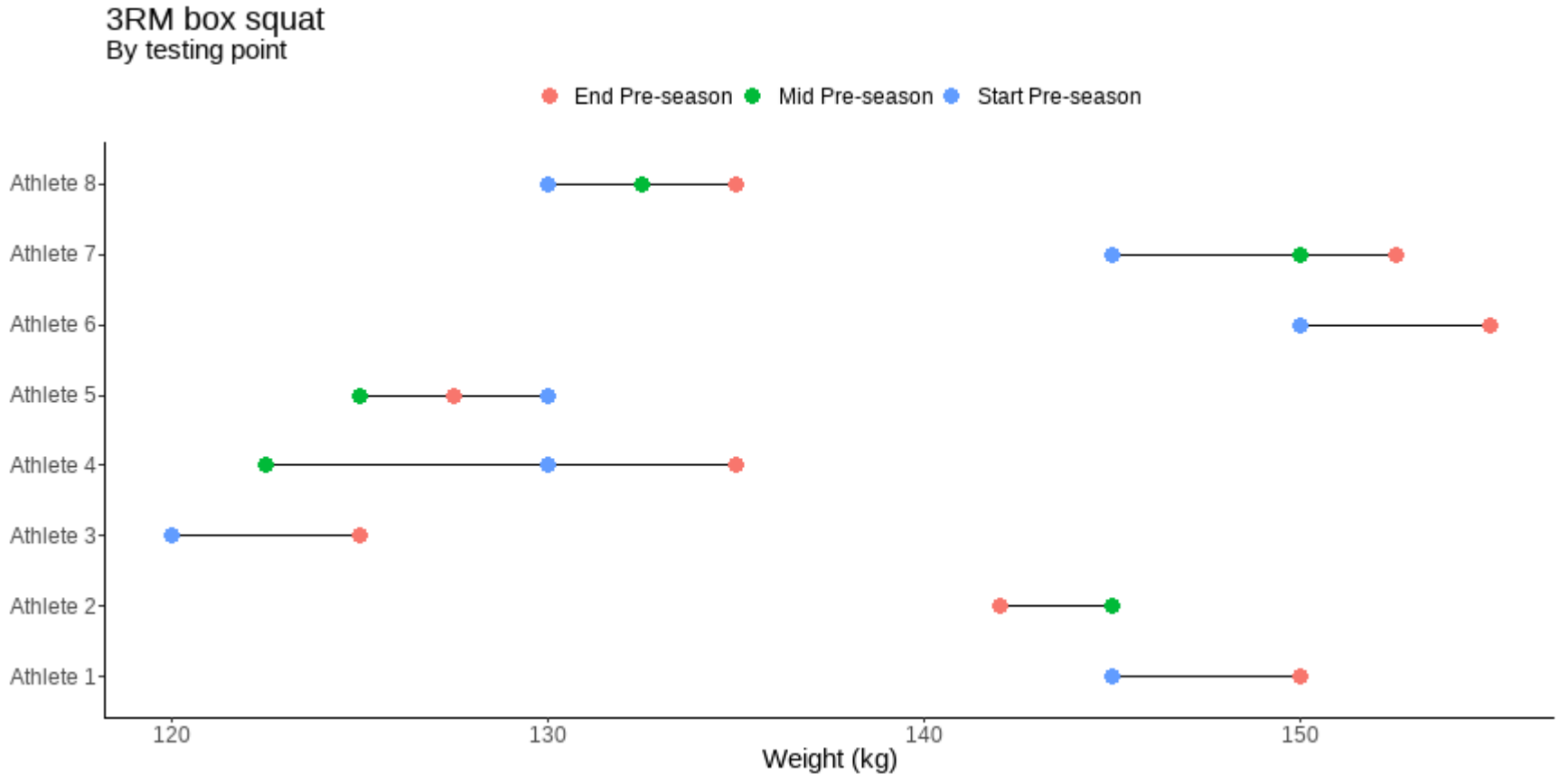
Name	Max chins (kg)	Change chins (kg)	Max bench (kg)	Change bench (kg)	Max squat (kg)	Change squat (kg)
Athlete 1	125.0	5.0	97.5	2.5	150.0	5.0
Athlete 2	127.5	2.5	107.5	10.0	145.0	3.0
Athlete 3	130.0	7.5	105.0	5.0	125.0	5.0
Athlete 4	127.5	5.5	115.0	5.0	135.0	12.5
Athlete 5	95.0	5.0	127.5	7.5	130.0	5.0
Athlete 6	117.5	7.5	107.5	2.5	155.0	5.0
Athlete 7	110.0	5.0	120.0	5.0	152.5	7.5
Athlete 8	115.0	2.5	100.0	2.5	135.0	5.0

Squat result by testing point

🏋 Create dumbell plot showing testing result by testing point

```
Strength %>%
            filter(Season == 2021) %>%

# Dumbell plot using geom_line and geom_point
ggplot(aes(x= `3RM Box Squat`, y= Name)) +

        geom_line(aes(group = Name))+ # add line between testing points
        geom_point(aes(color=Period), size=4) + # a dot for each testing point

        ggtitle("3RM box squat", subtitle = "By testing point") + # title
                xlab("Weight (kg)") + # axis label
        theme_classic()+ # simple theme
                theme(legend.position = "none", # hide legend
                      axis.title.y = element_blank()) # hide y axis title
```

Squat result by testing point



Example 2 - Nordboard data

Nordboard data

Import **Nordboard** data

```
Nordboard <- read_csv("Files/Nordboard.csv")
head(Nordboard, n=6)
```

```
## # A tibble: 6 x 7
##   Player    Date       RowIndex `Average Left F~ `Average Right ~ `Max Left Force`
##   <chr>     <chr>         <dbl>            <dbl>            <dbl>             <dbl>
## 1 Player 1  22/11/2019       10              394              415               405
## 2 Player 10 7/12/2020         6              414              399.              438
## 3 Player 10 15/12/2020        4              341.             331.              392.
## 4 Player 10 18/01/2021        2              498.             462.              508.
## 5 Player 10 3/02/2021         0              466.             439.              474.
## 6 Player 10 3/02/2021         2              459.             437.              471
## # ... with 1 more variable: Max Right Force <dbl>
```

Clean Nordboard data

- Add a new column for sets ('row index' is incorrect)

```
Nordboard <- Nordboard %>%
          group_by(Player, Date) %>%
            mutate(Set = seq_along(RowIndex))

# Then reorder variables
Nordboard <- Nordboard[c(1:2,8,4:7)]
```

Clean Nordboard data

➕ Add a new column for sets ('row index' is incorrect)

```
## # A tibble: 12 x 7
## # Groups:   Player, Date [10]
##    Player    Date         Set `Average Left F~ `Average Right ~ `Max Left Force`
##    <chr>     <chr>      <int>            <dbl>            <dbl>            <dbl>
##  1 Player 1  22/11/2019     1              394              415              405
##  2 Player 10 7/12/2020      1              414              399.             438
##  3 Player 10 15/12/2020     1              341.             331.             392.
##  4 Player 10 18/01/2021     1              498.             462.             508.
##  5 Player 10 3/02/2021      1              466.             439.             474.
##  6 Player 10 3/02/2021      2              459.             437.             471
##  7 Player 10 8/02/2021      1              455.             456              505
##  8 Player 10 8/02/2021      2              422.             446.             437.
##  9 Player 10 20/02/2021     1              406.             402.             422
## 10 Player 10 22/02/2021     1              414.             444.             422.
## 11 Player 10 2/03/2021      1              398.             426.             418.
## 12 Player 11 11/01/2020     1              375.             365.             399
## # ... with 1 more variable: Max Right Force <dbl>
```

Clean Nordboard data

↔ Now we need to rename columns. This will make it easier to differentiate left/right

```
# First we will rename a few columns to have the side (L/R) written first
Nordboard <- Nordboard %>%
  rename("Right_Average Force" = `Average Right Force`,
         "Left_Average Force" = `Average Left Force`,
         "Right_Max Force" = `Max Right Force`,
         "Left_Max Force" = `Max Left Force`)

head(Nordboard, n=5)
```

```
## # A tibble: 5 x 7
## # Groups:   Player, Date [5]
##   Player    Date         Set `Left_Average Fo~ `Right_Average ~ `Left_Max Force`
##   <chr>     <chr>      <int>             <dbl>             <dbl>             <dbl>
## 1 Player 1  22/11/2019     1               394               415               405
## 2 Player 10 7/12/2020      1               414               399.              438
## 3 Player 10 15/12/2020     1               341.              331.              392.
## 4 Player 10 18/01/2021     1               498.              462.              508.
## 5 Player 10 3/02/2021      1               466.              439.              474.
## # ... with 1 more variable: Right_Max Force <dbl>
```

Long to wide format

✖ When data is spread across columns, it can make it harder to visualise

✔ We can mimic what a **pivot table** might do in excel (but easier of course) using the **reshape2** package

Long format

Group	Value
A	12
B	10
C	8

Wide format

Group A	Group B	Group C
12	10	8

Long to wide format

Using the melt function in the **reshape2** package, we can flip the dataset by selected variables

```
library(reshape2)
Nordboard <- melt(Nordboard, id.vars = c("Player", "Date", "Set"))
```

```
##       Player       Date Set           variable    value
## 1   Player 1 22/11/2019   1 Left_Average Force 394.0000
## 2  Player 10  7/12/2020   1 Left_Average Force 414.0000
## 3  Player 10 15/12/2020   1 Left_Average Force 340.6500
## 4  Player 10 18/01/2021   1 Left_Average Force 497.8125
## 5  Player 10  3/02/2021   1 Left_Average Force 466.2500
## 6  Player 10  3/02/2021   2 Left_Average Force 458.9375
## 7  Player 10  8/02/2021   1 Left_Average Force 455.1250
## 8  Player 10  8/02/2021   2 Left_Average Force 422.5000
## 9  Player 10 20/02/2021   1 Left_Average Force 405.6875
## 10 Player 10 22/02/2021   1 Left_Average Force 413.6250
## 11 Player 10  2/03/2021   1 Left_Average Force 398.3750
## 12 Player 11 11/01/2020   1 Left_Average Force 374.9375
```

Add a new column (split text)

🖧 Now we want to split the 'variable' column into 2 so we have a 'leg' column

```
Nordboard <- Nordboard %>%
          separate(variable, into = c("Leg", "Variable"), sep = "_")
```

```
##       Player       Date Set  Leg      Variable    value
## 1   Player 1 22/11/2019   1 Left Average Force 394.0000
## 2  Player 10  7/12/2020   1 Left Average Force 414.0000
## 3  Player 10 15/12/2020   1 Left Average Force 340.6500
## 4  Player 10 18/01/2021   1 Left Average Force 497.8125
## 5  Player 10  3/02/2021   1 Left Average Force 466.2500
## 6  Player 10  3/02/2021   2 Left Average Force 458.9375
## 7  Player 10  8/02/2021   1 Left Average Force 455.1250
## 8  Player 10  8/02/2021   2 Left Average Force 422.5000
## 9  Player 10 20/02/2021   1 Left Average Force 405.6875
## 10 Player 10 22/02/2021   1 Left Average Force 413.6250
## 11 Player 10  2/03/2021   1 Left Average Force 398.3750
## 12 Player 11 11/01/2020   1 Left Average Force 374.9375
```

Visualise Nordboard data

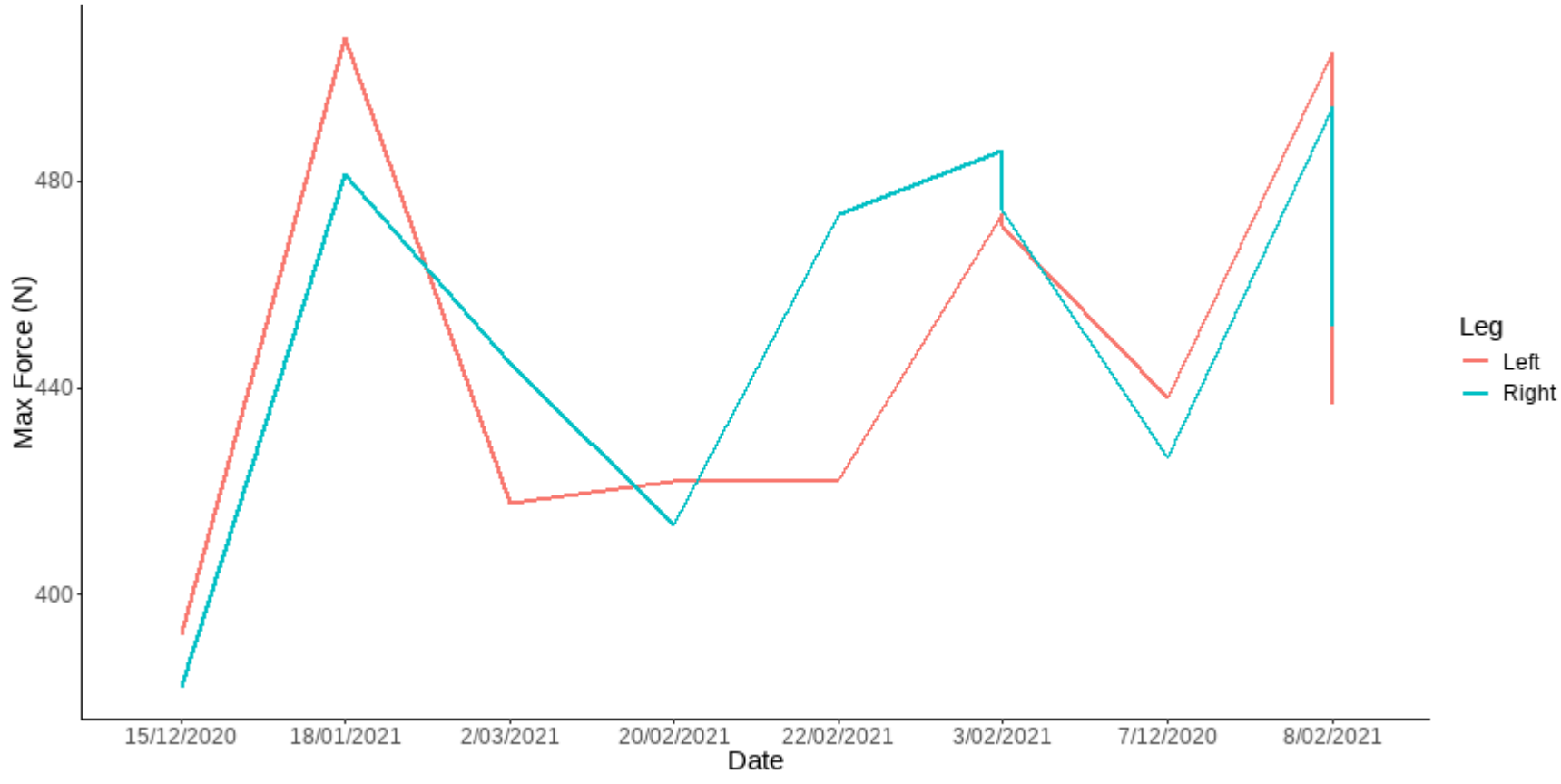
Now I want to filter by one athlete to visualise max force over time

```
filtNordboard <- Nordboard%>%
                  filter(Player == "Player 10") %>%
                  filter(Variable == "Max Force")
```

Using a powerful visualisation package called **ggplot2** we can create a basic line chart

```
library(ggplot2)
  ggplot(data = filtNordboard,
       aes(x=Date, y=value, group=Leg, colour = Leg)) +
        geom_line() +
        ylab("\n Max Force (N)") +
        theme_classic()
```

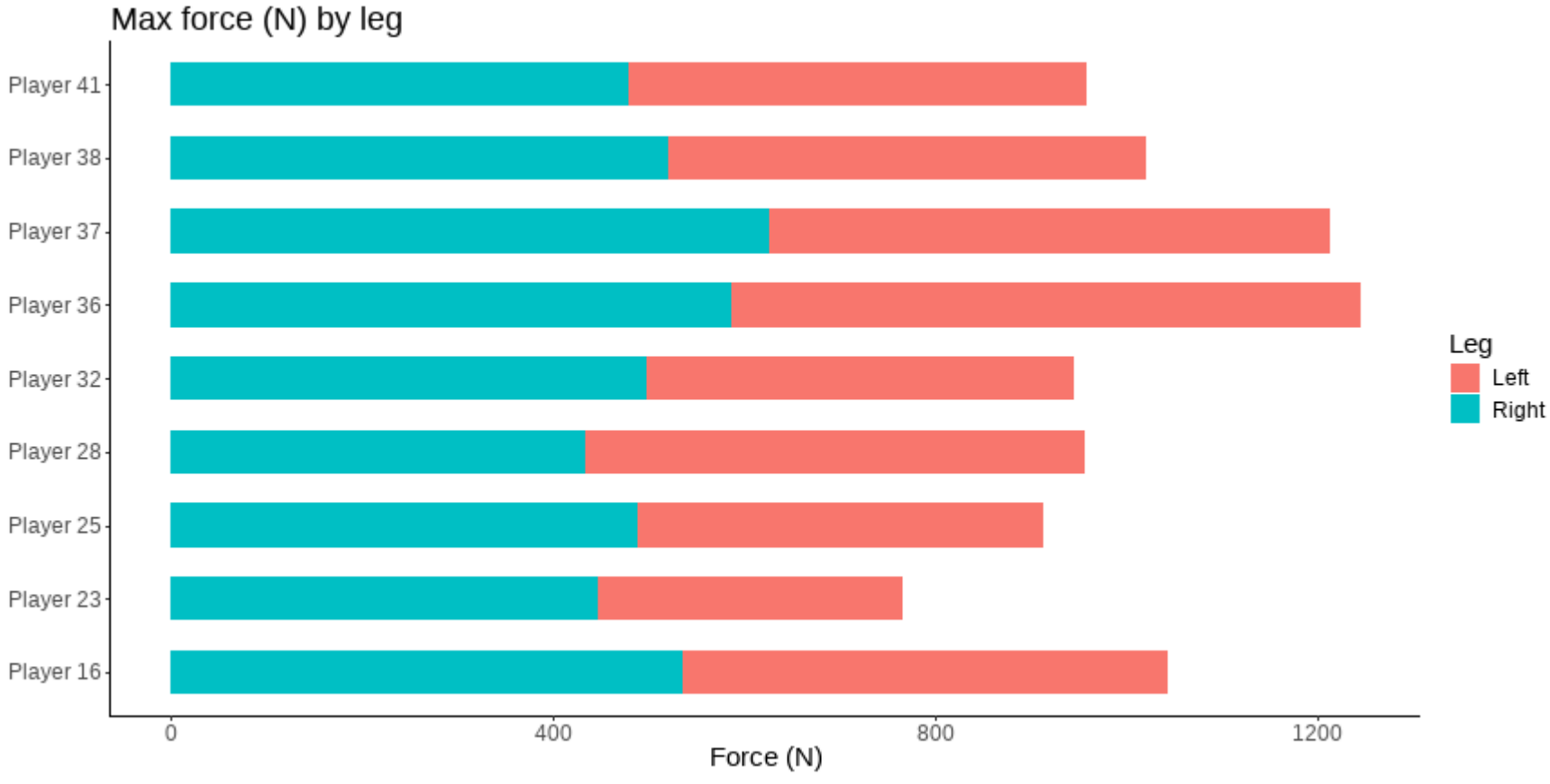
Visualise Nordboard data



Visualise Nordboard data

```
# Filter our data by date and max force
Nordboard %>%
            filter(Date == "10/03/2021") %>% filter(Variable == "Max Force") %>%
  # Summarise it by player and leg for the day selected above
                group_by(Player, Leg) %>%
  # Now calculate max force by leg by player
                    summarise(maxForce = max(value))  %>%

ggplot(aes(x = Player, y = maxForce, fill = Leg)) +
                          geom_bar(stat = "identity", width = .6) +
                          coord_flip() +
                          theme_classic() +
                          labs(title="Max force (N) by leg") + ylab("Force (N)") +
                          theme(axis.title.y = element_blank())
```

Visualise Nordboard data

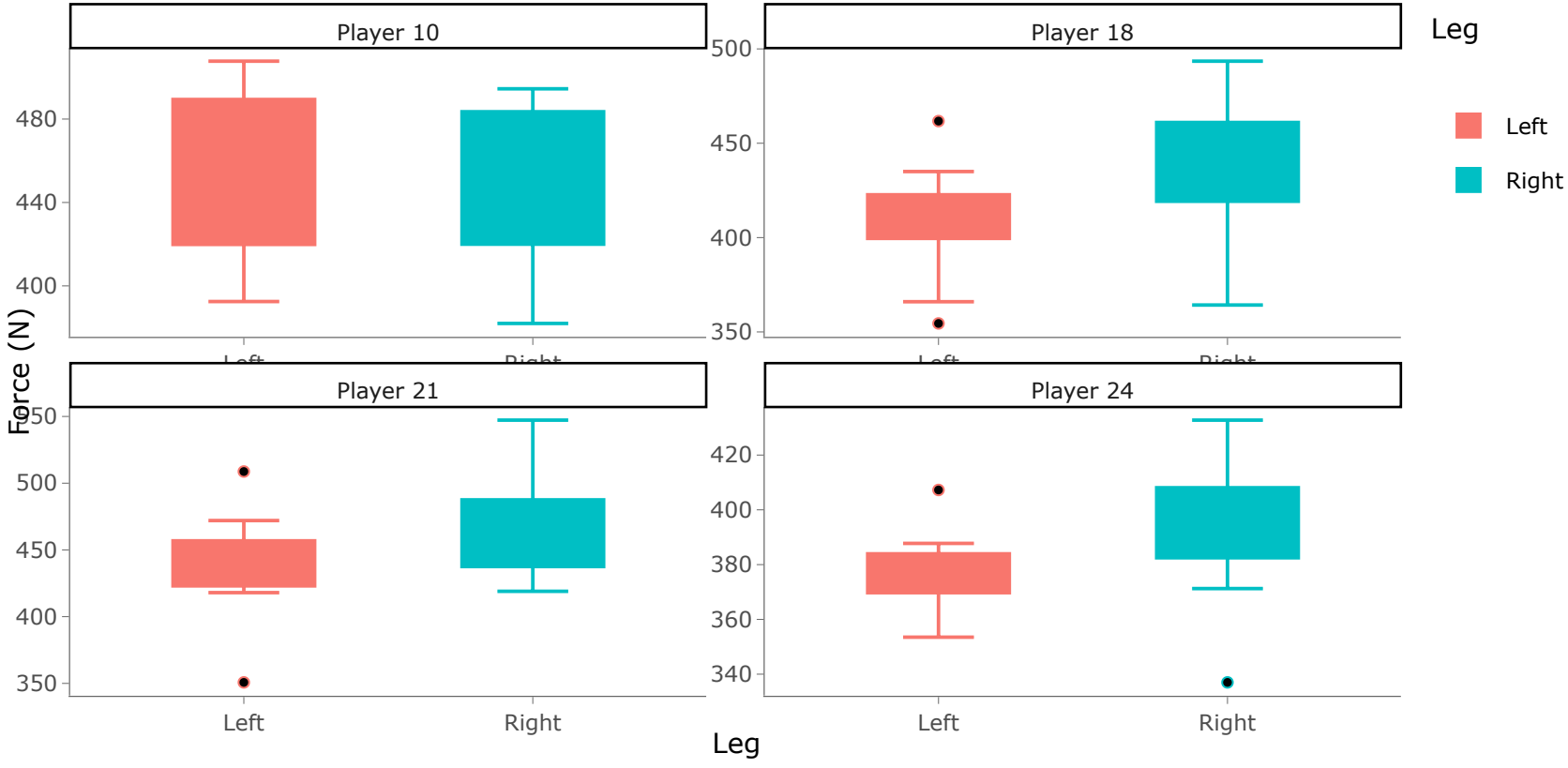


Visualise Nordboard data

Interactive box plots per athlete (by) using **plotly**

```
library(plotly)

target <- c("Player 10", "Player 21", "Player 24", "Player 18")

Nordboard1 <- Nordboard %>%
              filter(Variable == "Max Force") %>%
              filter(Player %in% target) %>%
                     group_by(Player, Leg, Date) %>%
                     summarise(maxForce = max(value))

plot <- ggplot(Nordboard1, aes(x = Leg, y=maxForce, fill = Leg, colour = Leg)) +
          geom_boxplot() +
          theme_classic() +
          ylab("Force (N)") +
          theme(axis.title.y = element_text(vjust=1)) +
          theme(panel.spacing = unit(1, "lines")) +
          facet_wrap(~Player, scales = "free")

fig <- ggplotly(plot)
fig
```

Visualise Nordboard data



Part 2 - Power BI

Why Power BI?

If you know how to use **Microsoft Excel**, you'll already know parts of **Microsoft Power BI**

Create simple, powerful visualisations, in a time efficient manner

$ It's **free** (desktop version)

All calculations/formulas can be done directly in **Power BI**

Download direct from the internet

Power BI exmaple - wellness report

Wellness report

👉 First, download **Power BI Desktop** from here

Example will show

- How to import data
- Create a table
- Add filters
- Create calculated columns
- Formatting

📈 Create wellness report inspired by Futbol AnalysR

Futbol AnalysR
@FutbolAnalysR

1/ Are you using a wellness monitoring dashboard with your team?

Does it start conversations with athletes or does it stop them from training?

#PowerBiDesktop #PowerBi #PowerBiForSportScience #PowerBiForSport #DAX #DAXMeasure #SportScience #WellnessMonitoring #Monitoring

Useful resources

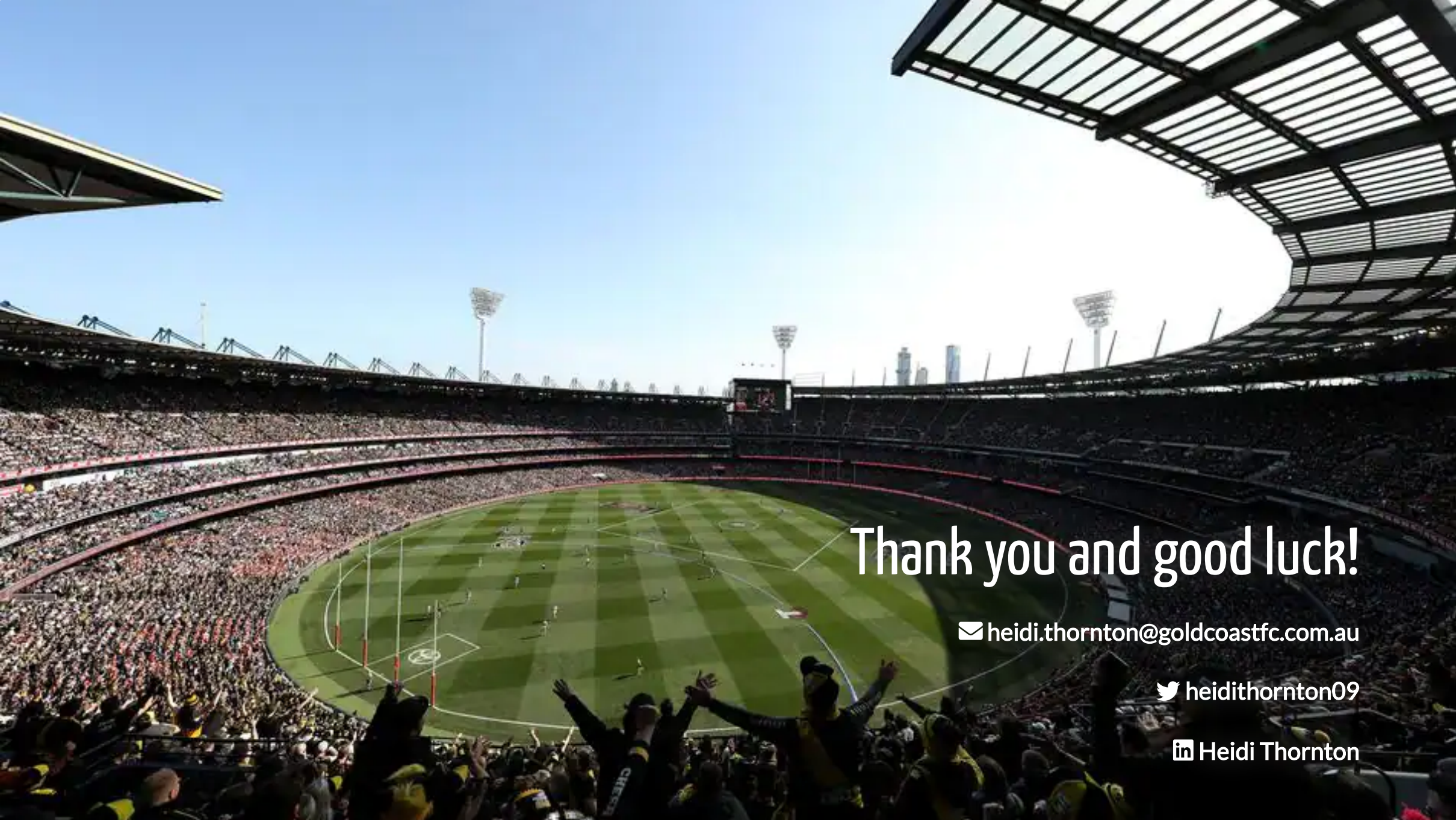
Below are links to personal websites/twitter pages/ youtube etc

R Studio

- Mitch Henderson
- Alice Sweeting
- Neil Collins
- Jose Fernandez

Power BI

- Josh Trewin
- Roberto D'Onofrio Rondón
- Complementary training



Thank you and good luck!

heidi.thornton@goldcoastfc.com.au

heidithornton09

Heidi Thornton