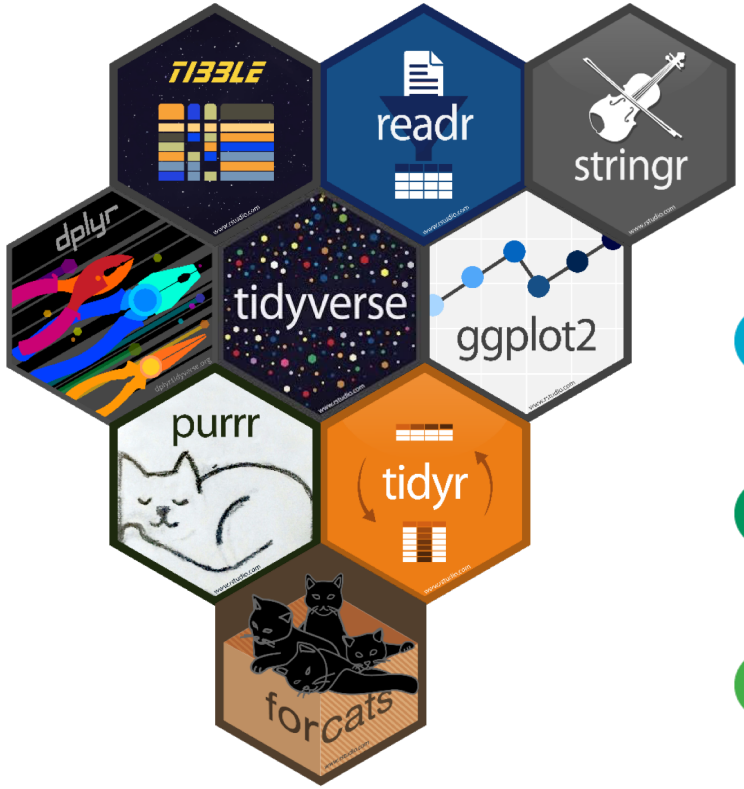
UK Data Service

Into the Tidyverse



Dr. J. Kasmire,
Research Fellow

Essex Summer School 2020

Day 2

What is the Tidyverse?

- An opinionated collection of R packages designed for data science.
- Does data exploration, manipulation, and visualisation
- Focus:
  - tibble as a structure
  - tidyr and dplyr for data manipulation
  - ggplot2 for visualisation

Tidyverse grammar

- Functions accept and return tibbles (opinionated data frames)
- String functions together with a pipe ‘ %>%’ (or R shortcut CTRL+SHIFT+M)
- Goal is readability and order
- Compare the following

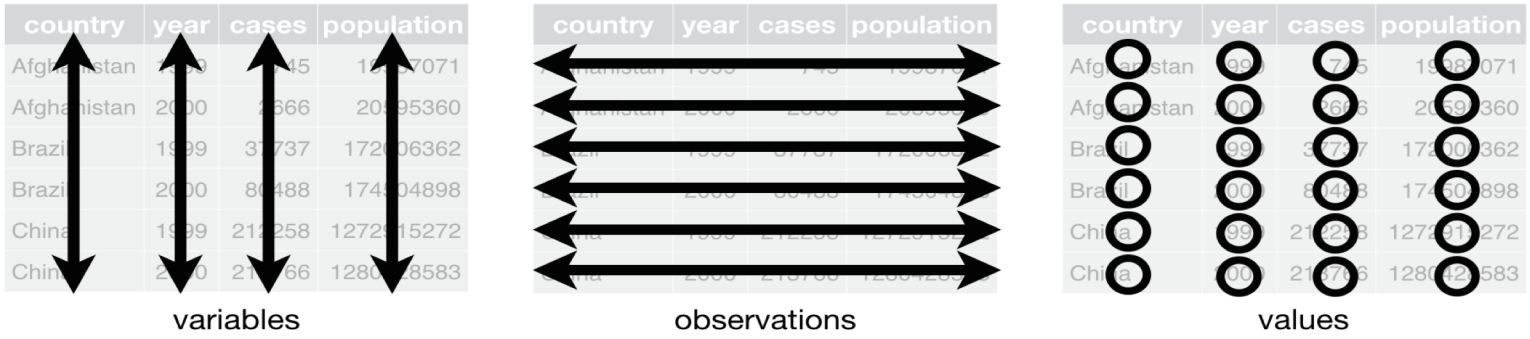
```
my_data_frame %>% group_by(UserID, Date, CategoryID)
                  %>% summarise(count=n())
```

```
tapply(rep(1, length.out=nrow(tab2)), INDEX=list(tab2$UserID, tab2$Date,
tab2$CategoryID),  FUN=sum)
```

UK Data Service

Tidyverse structure

- The same data can be tidy or not tidy
- Tidyverse functions are designed to work with tibbles and tidy data.
- Tidy data has:
  - Each observation must have its own row
  - Each variable must have its own column
  - Each value must have its own cell



UK Data Service

Tidyverse Exercises 1

Is it tidy?

- Each observation must have its own row
- Each variable must have its own column
- Each value must have its own cell

Country	Date	Incidents	Resolutions
UK	22/04/2020	18	10
France	22/04/2020	11	9
China	22/04/2020	21	18

Country	Date	Type	Counts
UK	22/04/2020	Incidents	18
France	22/04/2020	Incidents	11
China	22/04/2020	Incidents	21
UK	22/04/2020	Resolutions	10
France	22/04/2020	Resolutions	9
China	22/04/2020	Resolutions	18

Tibble – a tidy data frame

Tibbles vs. Classic data.frames

- Print
  - Prints type per column
  - Unless otherwise commanded, prints 10 rows and columns to fit display
- Subsetting
  - As value: tibble$column_name tibble[['column_name']]
    tibble[[1]] tibble[,1]
  - As variable = tibble[1] tibble[1,]
- Embed into a pipe with '.'
  - tibble %>% .$column_name or tibble %>% .[['column_name']] or tibble %>% .[[1]]

UK Data Service

Creating tibbles

Coerce existing data.frames into tibbles using:

```
data_name <- as_tibble(data.frame)
```

Create new tibbles using:

```
data_name <- tibble(columnA, columnB, columnN)
```

Tibble exercises

Read Mammals.csv, Population_data.txt and Suicides.xlsx into R.

Print one of them.

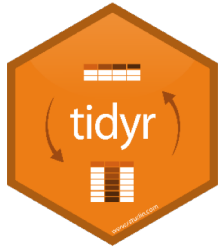
Convert all three to tibbles and print the same one as before.

Extract a column as a value in 2 different ways.

Extract one column as a variable.

Extract a row as a variable.

Tidyr – make it tidy



- A dedicated tool to get data into a tidy format
- 3 main functions:
  - gather() - can put each observation in its own row
  - spread() – can put each variable in its own column
  - separate() – can put each value in its own cell

UK Data Service

gather() – turns wide data into long data

heroes %>% gather(Universe, Name, Marvel:DarkHorseComics)

Gender	Marvel	DC Comics	Dark Horse Comics
Male	Ant-Man	Green Arrow	Hellboy
Female	Elektra	Zatanna	Ghost

Gender	Universe	Name
Male	Marvel	Ant-Man
Female	Marvel	Elektra
Male	DC Comics	Green Arrow
Female	DC Comics	Zatanna
Male	Dark Horse Comics	Hellboy
Female	Dark Horse Comics	Ghost

gather() – turns wide data into long data

heroes %>% gather(Universe, Name, Marvel:DarkHorseComics)

Name of column to hold gathered column headings

Name of column to hold gathered column data

Gender	Marvel	DC Comics	Dark Horse Comics
Male	Ant-Man	Green Arrow	Hellboy
Female	Elektra	Zatanna	Ghost

The columns with data to be gathered

Gender	Universe	Nam
Male	Marvel	Ant-Man
Female	Marvel	Elektra
Male	DC Comics	Green Arrow
Female	DC Comics	Zatanna
Male	Dark Horse Comics	Hellboy
Female	Dark Horse Comics	Ghost

UK Data Service

spread() – turns long data into wide data

heroes %>% spread(Universe, Name)

Gender	Marvel	DC Comics	Dark Horse Comics
Male	Ant-Man	Green Arrow	Hellboy
Female	Elektra	Zatanna	Ghost

Gender	Universe	Name
Male	Marvel	Ant-Man
Female	Marvel	Elektra
Male	DC Comics	Green Arrow
Female	DC Comics	Zatanna
Male	Dark Horse Comics	Hellboy
Female	Dark Horse Comics	Ghost

spread() – turns wide data into long data

heroes %>% spread(Universe, Name)

Name of column with intended column headings

Name of column with intended column data

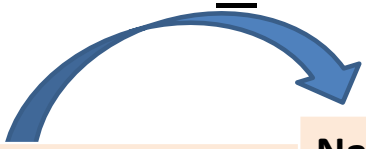
Gender	Marvel	DC Comics	Dark Horse Comics
Male	Ant-Man	Green Arrow	Hellboy
Female	Elektra	Zatanna	Ghost

Gender	Universe	Na…
Male	Marvel	Ant-Man
Female	Marvel	Elektra
Male	DC Comics	Green Arrow
Female	DC Comics	Zatanna
Male	Dark Horse Comics	Hellboy
Female	Dark Horse Comics	Ghost

separate() – separates multi-value cells

heroes %>% spread(Gender_film, c("Gender", "Has_film"), ",")

Name	Gender_film	Universe
Ant-Man	Male,1	Marvel
Elektra	Female,1	Marvel
Green Arrow	Male,0	DC Comics
Zatanna	Female,0	DC Comics
Hellboy	Male,1	Dark Horse Comics
Ghost	Female,0	Dark Horse Comics



Name	Gender	Has_film	Universe
Ant-Man	Male	1	Marvel
Elektra	Female	1	Marvel
Green Arrow	Male	0	DC Comics
Zatanna	Female	0	DC Comics
Hellboy	Male	1	Dark Horse Comics
Ghost	Female	0	Dark Horse Comics

separate() – splits multi-value cells

heroes %>% spread(Gender_film, c("Gender", "Has_film"), ",")

Name of the column to be split

Name of columns to store split values into

Name	Gender_film	Universe
Ant-Man	Male,1	Marvel
Elektra	Female,1	Marvel
Green Arrow	Male,0	DC Comics
Zatanna	Female,0	DC Comics
Hellboy	Male,1	Dark Horse Comics
Ghost	Female,0	Dark Horse Comics

Where to split

If a character, values are split AT that character

Name	Gender	Has_film	Universe
Ant-Man	Male	1	Marvel
Elektra	Female	1	Marvel
Green Arrow	Male	0	DC Comics
Zatanna	Female	0	DC Comics
Hellboy	Male	1	Dark Horse Comics
Ghost	Female	0	Dark Horse Comics

UK Data Service

separate() – splits multi-value cells

heroes %>% spread(Gender_film, c("Gender", "Has_film"), "2")

Name	Gender_film	Universe
Ant-Man	M,1	Marvel
Elektra	F,1	Marvel
Green Arrow	M,0	DC Comics
Zatanna	F,0	DC Comics
Hellboy	M,1	Dark Horse Comics
Ghost	F,0	Dark Horse Comics

Where to split
If a character, split AT that character
If numeric, split AFTER numerical position

Name of the column to be split

Name of columns to store split values into

Name	Gender	Has_film	Universe
Ant-Man	M	1	Marvel
Elektra	F	1	Marvel
Green Arrow	M	0	DC Comics
Zatanna	F	0	DC Comics
Hellboy	M	1	Dark Horse Comics
Ghost	F	0	Dark Horse Comics

UK Data Service

Exercises: tidyr

Use the tidyr functions in order to get our three tibbles into tidy form.

Each will need one or more functions.

UK Data Service

dplyr

Once tidy, you can manipulate your data with dplyr

We introduce:

- select()
- arrange()
- group_by()
- summarise() or summarize()
- mutate()
- join()

UK Data Service

select() – select subset of columns

select(heros, Name, Gender, Universe)
select(heros, Name:Gender, Universe)
select(heros, -(Has_film))

Name	Gender	Has_film	Universe
Ant-Man	Male	1	Marvel
Elektra	Female	1	Marvel
Green Arrow	Male	0	DC Comics
Zatanna	Female	0	DC Comics
Hellboy	Male	1	Dark Horse Comics
Ghost	Female	0	Dark Horse Comics

Name	Gender	Universe
Ant-Man	Male	Marvel
Elektra	Female	Marvel
Green Arrow	Male	DC Comics
Zatanna	Female	DC Comics
Hellboy	Male	Dark Horse Comics
Ghost	Female	Dark Horse Comics

filter() – select subset of rows

filter(heros, Gender == ‘Female’)
filter(heros, Gender == ‘Female’, Has_film == 1)

Name	Gender	Has_film	Universe
Ant-Man	Male	1	Marvel
Elektra	Female	1	Marvel
Green Arrow	Male	0	DC Comics
Zatanna	Female	0	DC Comics
Hellboy	Male	1	Dark Horse Comics
Ghost	Female	0	Dark Horse Comics

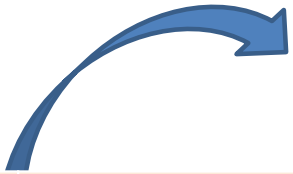
Name	Gender	Has_film	Universe
Elektra	Female	1	Marvel
Zatanna	Female	0	DC Comics
Ghost	Female	0	Dark Horse Comics

Name	Gender	Has_film	Universe
Elektra	Female	1	Marvel

UK Data Service

arrange() – reorder rows

arrange(heros, Gender, Universe)



Name	Gender	Has_film	Universe
Ant-Man	Male	1	Marvel
Elektra	Female	1	Marvel
Green Arrow	Male	0	DC Comics
Zatanna	Female	0	DC Comics
Hellboy	Male	1	Dark Horse Comics
Ghost	Female	0	Dark Horse Comics

Name	Gender	Has_film	Universe
Ghost	Female	0	Dark Horse Comics
Zatanna	Female	0	DC Comics
Elektra	Female	1	Marvel
Hellboy	Male	1	Dark Horse Comics
Green Arrow	Male	0	DC Comics
Ant-Man	Male	1	Marvel

UK Data Service

summarise() – summarises data

summarise(heros, mean(Start_date))

Takes a vector, returns a single value

Name	Gender	Has_film	Start_date
Ant-Man	Male	1	1962
Elektra	Female	1	1982
Green Arrow	Male	0	1941
Zatanna	Female	0	1964
Hellboy	Male	1	1993
Ghost	Female	0	1995

summarise() + group_by() = group summaries

```
Heros_grouped <- group_by(heros, Gender)
summarise(heros_grouped, mean(Start_date))
```

Takes a

summary per group

Name	Gender	Has_film	Start_date
Ant-Man	Male	1	1962
Elektra	Female	1	1982
Green Arrow	Male	0	1941
Zatanna	Female	0	1964
Hellboy	Male	1	1993
Ghost	Female	0	1995

Male 1965

UK Data Service

mutate() – alters data

```
heros %>%   filter(has_film == '1') %>%  mutate(Film_delay =
Film_date - Start_date)
```

Takes vector(s), returns a vector

Name	Start_date	Film_date
Ant-Man	1962	2015
Elektra	1982	2005
Green Arrow	1941	
Zatanna	1964	
Hellboy	1993	2004
Ghost	1995	

Name	Start_date	Film_date	Film_delay
Ant-Man	1962	2015	53
Elektra	1982	2005	24
Hellboy	1993	2004	11

UK Data Service

Exercises: dplyr

Using the tidy suicides data

- calculate the number of suicides per year
- add a new column to the data which contains the rate of suicide per 100,000 population

Read in the file “AMT_new.xlsx” and use the dplyr functions to calculate the number of applications made in each region (column AdaptedNUTS)

- store the result in an object called AMT_counts

Merging data sets

- There are several options in dplyr for merging data
  - These are called joins
    1. inner_join()
    2. left_join()
    3. right_join()
    4. full_join()
    5. semi_join()
    6. anti_join()
- These take two data frames, x and y and join by some variable

UK Data Service

Inner Join

- Creates a linked table with rows from each table that have a match

inner_join(Gender, Universe, by='Name')

Name	Gender
Ant-Man	Male
Elektra	Female
Green Arrow	Male
Zatanna	Female

Name	Universe
Green Arrow	DC Comics
Zatanna	DC Comics
Hellboy	Dark Horse Comics
Ghost	Dark Horse Comics

Name	Gender	Universe
Green Arrow	Male	DC Comics
Zatanna	Female	DC Comics

Left Join

- Keeps all of first table, ignores second table unless it has a match

left_join(Gender, Universe, by='Name')

Name	Gender
Ant-Man	Male
Elektra	Female
Green Arrow	Male
Zatanna	Female

Name	Universe
Green Arrow	DC Comics
Zatanna	DC Comics
Hellboy	Dark Horse Comics
Ghost	Dark Horse Comics

Name	Gender	Universe
Ant-Man	Male	NA
Elektra	Female	NA
Green Arrow	Male	DC Comics
Zatanna	Female	DC Comics

UK Data Service

Right Join

- Keeps all of second table, ignores first table unless it has a match

right_join(Gender, Universe, by='Name')

Name	Gender
Ant-Man	Male
Elektra	Female
Green Arrow	Male
Zatanna	Female

Name	Universe
Green Arrow	DC Comics
Zatanna	DC Comics
Hellboy	Dark Horse Comics
Ghost	Dark Horse Comics

Name	Gender	Universe
Green Arrow	Male	DC Comics
Zatanna	Female	DC Comics
Hellboy	NA	Dark Horse Comics
Ghost	NA	Dark Horse Comics

Full Join

- Keeps everything in both tables

full_join(Gender, Universe, by='Name')

Name	Gender
Ant-Man	Male
Elektra	Female
Green Arrow	Male
Zatanna	Female

Name	Universe
Green Arrow	DC Comics
Zatanna	DC Comics
Hellboy	Dark Horse Comics
Ghost	Dark Horse Comics

Name	Gender	Universe
Ant-Man	Male	NA
Elektra	Female	NA
Green Arrow	Male	DC Comics
Zatanna	Female	DC Comics
Hellboy	NA	Dark Horse Comics
Ghost	NA	Dark Horse Comics

Semi Join

- Keeps rows AND INFO from first table that have a match in second

semi_join(Gender, Universe, by='Name')

Name	Gender
Ant-Man	Male
Elektra	Female
Green Arrow	Male
Zatanna	Female

Name	Universe
Green Arrow	DC Comics
Zatanna	DC Comics
Hellboy	Dark Horse Comics
Ghost	Dark Horse Comics

Name	Gender
Green Arrow	Male
Zatanna	Female

Anti Join

- Keeps rows & INFO from 1st table that DO NOT have match in 2nd

anti_join(Gender, Universe, by='Name')

Name	Gender
Ant-Man	Male
Elektra	Female
Green Arrow	Male
Zatanna	Female

Name	Universe
Green Arrow	DC Comics
Zatanna	DC Comics
Hellboy	Dark Horse Comics
Ghost	Dark Horse Comics

Name	Gender
Ant-Man	Male
Elektra	Female

UK Data Service

Exercises: Merging data sets

- load the workspace “joining.RData” then using the join functions in dplyr:
  - Find all of the rows in Expenditure which have a match in Production
  - merge Production and Employees, keeping all rows in Employees
  - merge Employees and Expenditure, keeping all rows
  - Find the rows in Production which do not have a match in Employees

- load the workspace “mapping.RData”
  - The object “mapdata” contains shape information on all of the regions in the object “AMT_counts”. The object “link” links the two objects.
  - Use the join functions to merge the “mapping” object with the object of counts created during the last set of exercises

ggplot2 - create plots in layers

Call ggplot() first, followed by any number of geometries

- geom_hist()
- geom_bar()
- geom_label()
- geom_point()
- geom_line()
- geom_polygon()
- Plus more…

Each of these requires a mapping specified via the aes() function

UK Data Service

Aesthetic Mapping

The aes() function takes as arguments the things you want to plot

- For a histogram of bar chart, this would be one variable in your data
- For a scatter plot this would be two variables corresponding to the x and y axes

The aes() function also takes any specifications for colour, shape, style, etc. that use data.

Combining layers

You can add as many plot layers as you wish with the + sign

Example: a scatter plot with a line of best fit

```
ggplot(data) +
geom_point(aes(x,y)) +
geom_abline(aes(intercept=beta0, slope=beta1))
```

Use ANY of the geometries used in ggplot2
Try to only combine geometries that make sense

Exercises: ggplot2

Create a scatterplot of rem sleep against total sleep in the mammals data set

- add the fitted regression line to this plot
- colour the points by the “vore” variable

Create a histogram of the rate variable that you created in the suicides data

Plot the regions in the “mapdata” data

- colour these by the number of applications made

Lubridate and Zoo

The lubridate and zoo packages can help us work with time and date data in R.

Zoo can be used to convert data into ‘zoo’ series – these are ordered indexed observations. This applies a date to the observations, but importantly can be used when the dates required are irregular.

Lubridate can be used to combined variables together in order to create a time and date variable. For example, if a dataset contained separate columns for day, month, and year, lubridate would enable us to combine these into a date variable.

Exercise: Lubridate

Import the csv file ‘coronavirus-cases.csv’ into R.

Load the Lubridate package.

We will be using the Year, Month, and Day columns to produce a Date-class variable.

Lubridate allows us to produce a date in many formats using dmy(), myd(), ymd(), ydm(), dym(), mdy()

Create a new column named ‘date’ in the date format you wish

e.g. Covid$Date <- dmy(paste(Covid$Day, Covid$Month, Covid$Year))

UK Data Service

Exercise: Zoo

Load the zoo and dygraphs packages

Use the select function to create a new dataframe containing only the column ‘Daily cases’ from the data we previously imported.

Now we will use the zoo function to convert our new dataframe into a zoo series. The data we are using was first collected on the 1st March and runs until the 19th July, though it is in ‘reverse’ order.

```
Covid2 <- zoo(Covid2, seq(from = as.Date("2020-07-19"), to = as.Date("2020-03-01"), by = -1))
```

Now we will plot the data using the dygraphs package:

```
dygraph(Covid2, main = "Covid-19 Daily Cases in the UK") %>%
dyRangeSelector(dateWindow = c("2020-03-01", "2020-07-19"))
```

UK Data Service