Welcome to the tidyverse

…a comparison with the base package

Loading from File with Base Package

Do it yourself:

```
> arctic_df <- read.csv("NOAA_Arctic.csv") ↲
```

What went wrong? Go look at the file.

Take a look at the CSV file using a text editor. What do you see?

Now try:

```
> arctic_df <- read.csv("NOAA_Arctic.csv",
comment.char = "#")↲
```

Other useful arguments are skip and header.

Loading from File with tidyverse

First ~~load the library~~ (if tidyverse not loaded):

```
> library(tidyverse) ↲
```

We might have loaded just the `readr` package but here we load all the tidyverse metapackage. Now let's use readr to load the data into a `tibble`:

Do it yourself:

```
> arctic <- read_csv("NOAA_Arctic.csv", comment
= "#")↲
```

Notice the different arguments.

So what? Look carefully at the column names. Notice the use of `` around `complex column names`.

The dplyr paradigm 1

Now we can do several things to the tibble (using dplyr, also in the tidyverse):

Do it yourself:

```
> filter(arctic, Month==1, Day==1)↵

> filter(arctic, `Extent (10^6 sq km)`<=3.5) ↵
```

Notice the different arguments.

Here you are selecting rows based on values. It is also possible to use & (for AND) and | (for OR) queries.

The dplyr paradigm 2

Do it yourself: let's tame the lengthy variable name:

```
> arctic <- rename(arctic, Extent=`Extent (10^6 sq km)`) ↵
```

Now let's try these:

```
> select(arctic, Year) ↵
# equivalent to arctic$Year
> select(arctic, Year:Day)
```

Now make a new variable:

```
> mutate(arctic, logex=log(Extent))
```

what happens if you use transmute instead of mutate?

Summarising Data 1

Summarise[1] can be used to convert to a single row:

```
> summarise(arctic, area=min(Extent)) ↵
```

More useful if you can group the data:

```
> by_year <- group_by(arctic, Year)↵
# go take a look at this object:
> by_year ↵
# doesn't look very interesting – but let's summarise that instead
> min_ice <- summarise(by_year,area=min(Extent))
# now what's happened?
> min_ice ↵
```

We are making a lot of intermediate objects here and having to inspect them

[1] yes UK spelling here – tidyverse author is from New Zealand…

Summarising Data 2

Let's do this slightly differently:

```
> ( max_ice <-
summarise(by_year,area=max(Extent)) ) ↵
```

Wrapping an assignment in outer brackets prints output to the screen as well as assigning.

So we can also look at it whenever we wish:

```
> max_ice ↵
```

But we are still working with intermediate objects. That could get untidy.

Piping Data

Using pipes:

```
> ( lean_yrs <- arctic %>% ↲
+ group_by(Year) %>% ↲
+ summarise(area=min(Extent)) %>% ↲
+ filter(area < 5) ) ↲
```

Using pipes enables you to skip references and pass your analyses on for further processing.

In this example, we piped from line to line but you can enter all in one line.

So, I highly recommend consulting R for Data Science, in particular: chapters 5 and 12

but…

enough already give me a graph!

Ok, but then we'll have to go slow again…

All on one slide!

Using ggplot2 (part of tidyverse):

```
library(lubridate)
arctic %>%
mutate(Date2=make_datetime(Year, Month, Day)) %>%
mutate(DayinYear=as.numeric(format(Date2, "%j"))) %>%
ggplot(aes(x=DayinYear, y=Extent, colour=Year)) +
geom_point()
```

But we'll come back to this package later…