

# Data Visualisation

Data Science  
Week 4

# Today's class

General Things...

First of all, reminder about AI

Part 1: Making Plots and Intro to ggplot

- Lecture + Class Activity

Part 2: Communicating with Data

- “Impress my Nieces!”

# General things...

- Remember to keep engaging on Github! (There's no way so few people are encountering issues...)
- Re cloning the course repo...
- If anyone is struggling – talk to us!!
- You will struggle in these classes without coding club/data camp, make sure to engage with the online lessons

# Clean-That-Code reminder

- Finish cleaning your code by Tuesday (14<sup>th</sup>)
- Begin giving feedback to your partner from NEXT Wednesday (15<sup>th</sup>)
- Everything due on the 17<sup>th</sup>, at which point we'll release the answers

AI!

# Data Visualisation

# Data Visualisation

<https://medium.com/bbc-visual-and-data-journalism/how-the-bbc-visual-and-data-journalism-team-works-with-graphics-in-r-ed0b35693535>

<https://infographics.economist.com/2021/job-data-visualisation-trainee/index.html>

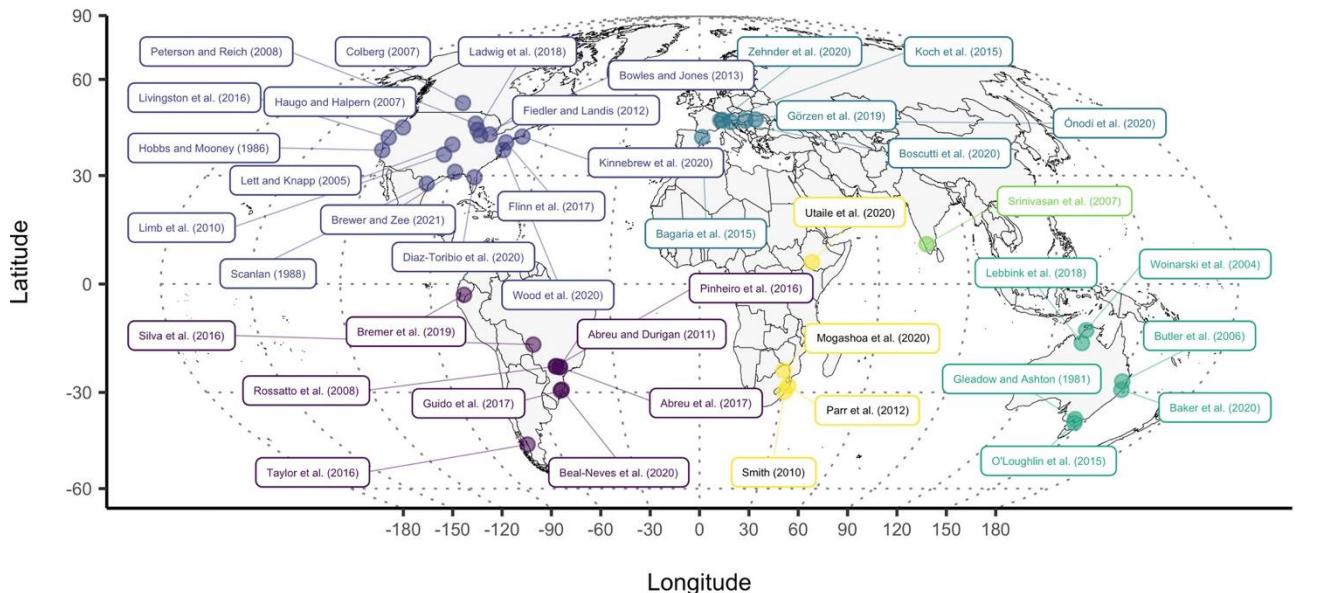
# What makes a good plot?

# What makes a good plot?

- Easy to see the overall message
- Stylistic, clean lines, not cluttered
- Good use of colour (inclusivity!), *only when necessary*
- Represents the data

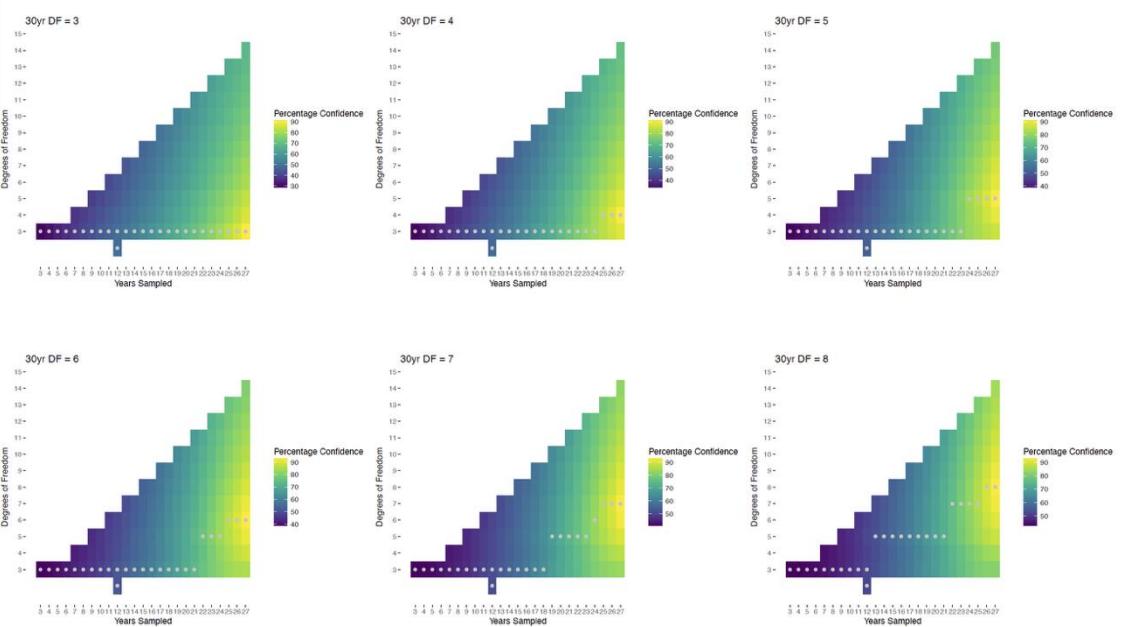
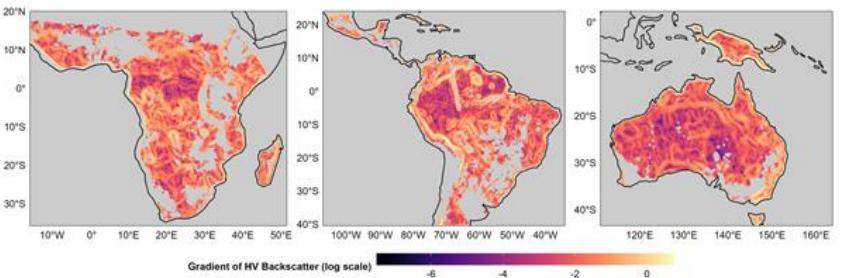
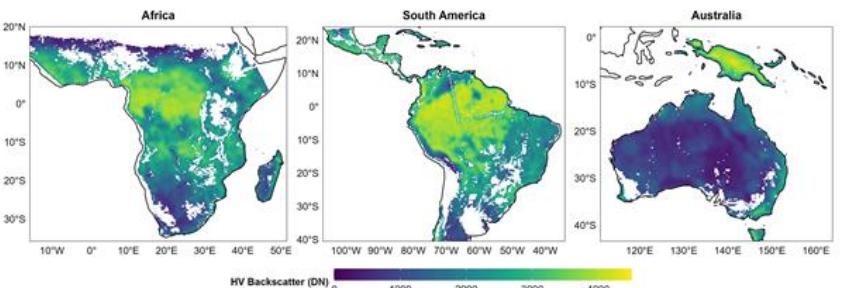
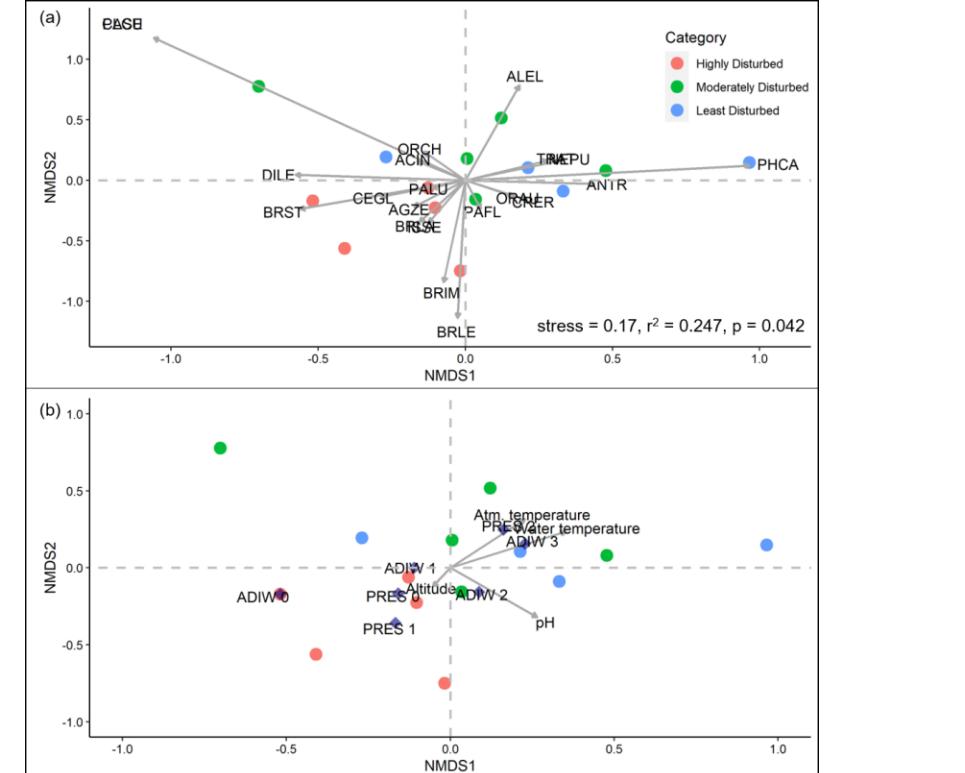
What do we think of these plots....

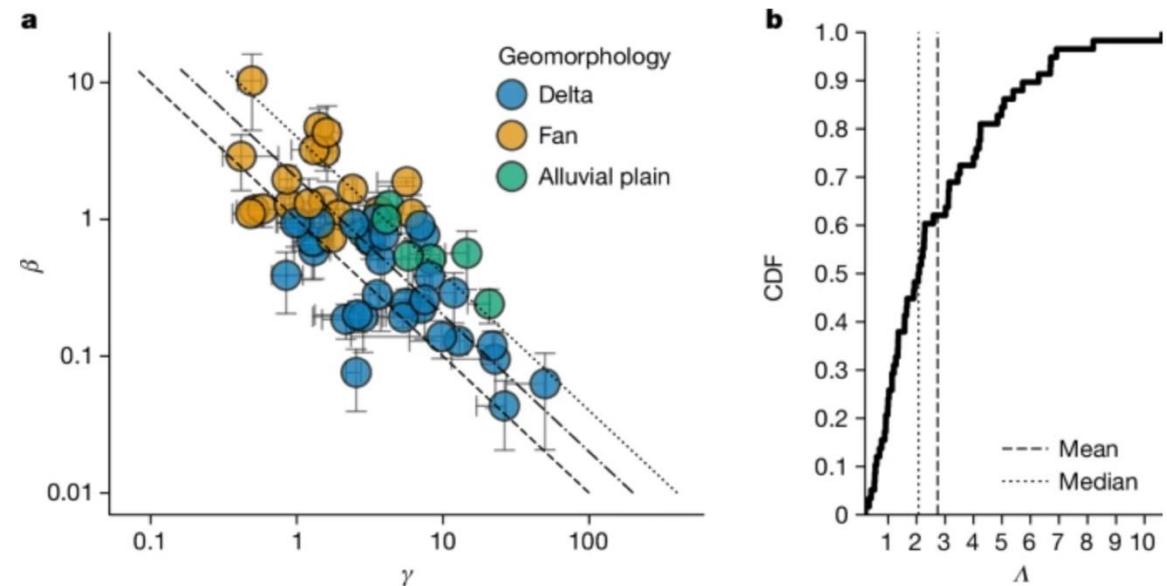
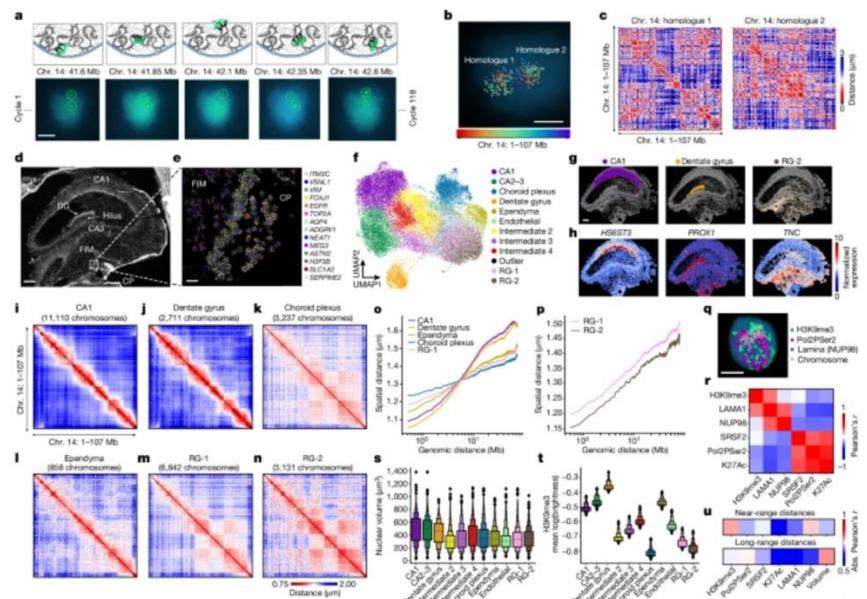
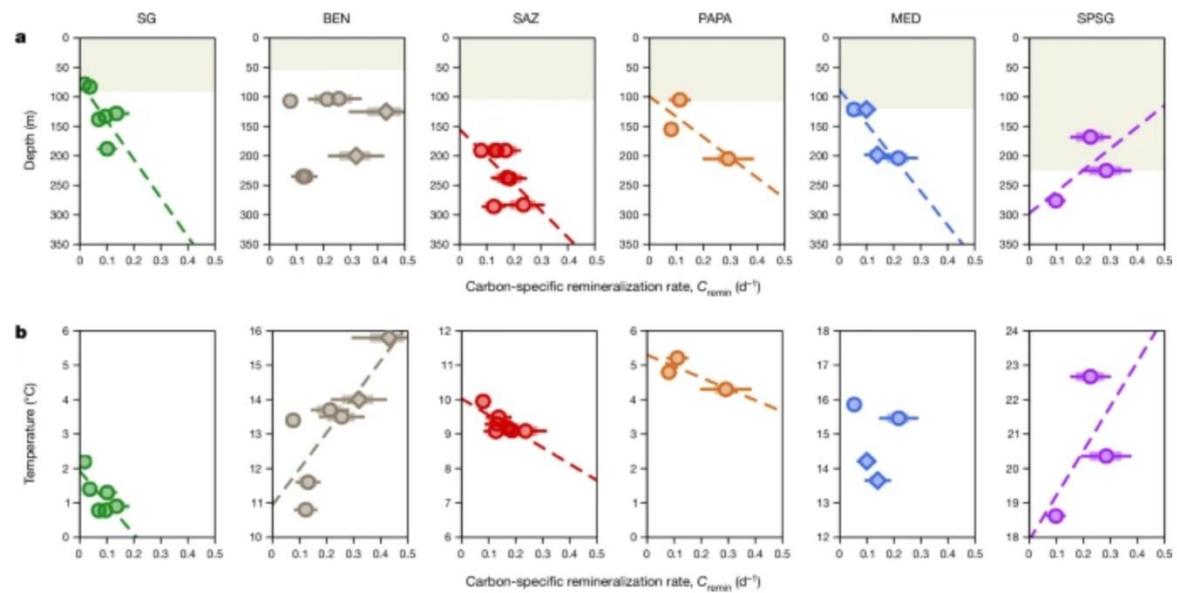
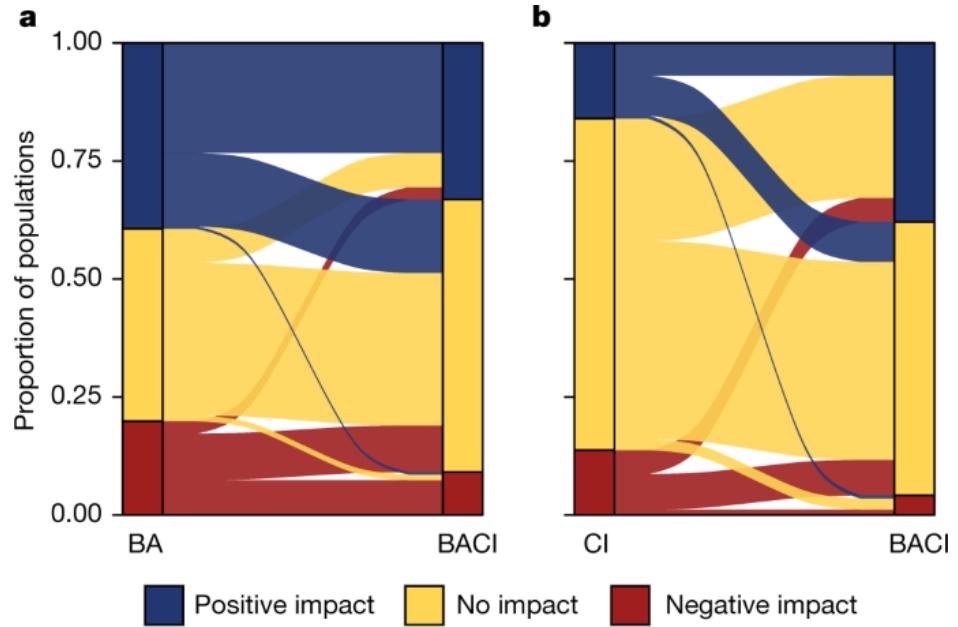
# Locations of study sites



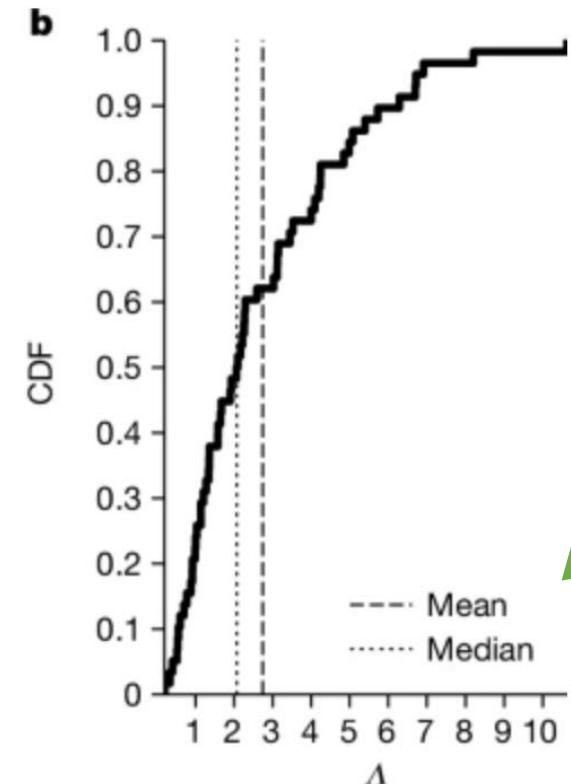
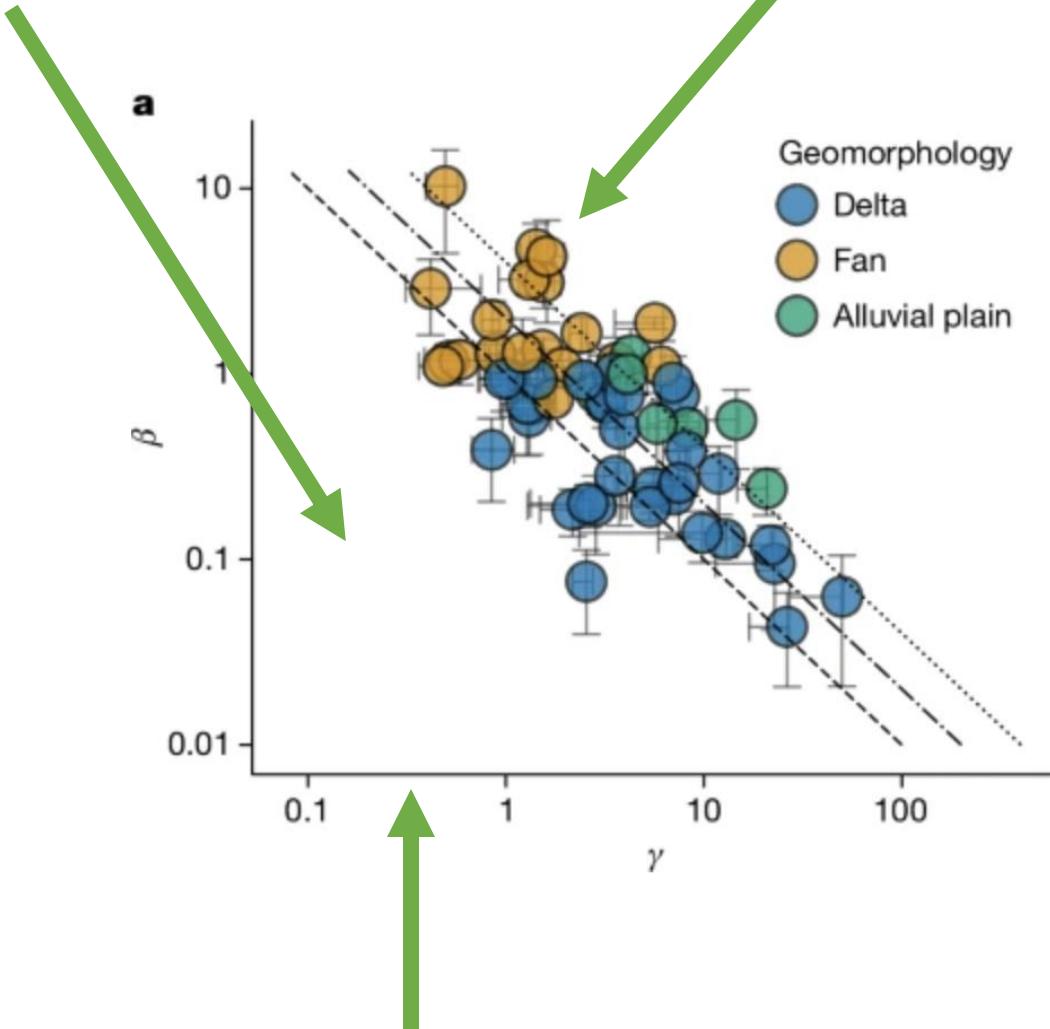
## Continent

- Africa
- Asia
- Australia
- Europe
- N. America
- S. America





No gridlines, white background Muted primary colours



Clean black lines

Plot ratio aligns with data

ggplot

# ggplot basics

- ggplot works by stacking various commands on top of each other.
- E.g. what data to use
- E.g. what ‘layers’ to plot (e.g. points, boxplots)
- E.g. how various aspects of the graph should look.

# ggplot basics

- First, we start with the data and the axes

```
ggplot(data=YourData, aes(x = xaxis, y = yaxis))+
```



“aes” brackets anything you want to vary on the plot

# ggplot basics

- First, we start with the data and the axes

```
ggplot(data=YourData, aes(x = xaxis, y = yaxis))+
```



Put a plus sign at the end of  
every line to tell R you're still  
building (until last line)

# ggplot basics

- Next, we add what we want to plot on the graph

```
ggplot(data=YourData, aes( x = xaxis, y = yaxis))+  
  geom_point()
```

# ggplot basics

- Next, maybe we want to adjust the x axis

```
ggplot(data=YourData, aes( x = xaxis, y = yaxis))+  
  geom_point() +  
  scale_x_continuous(...)+
```

# ggplot basics

- Finally, we want to adjust how the graph looks aesthetically, using theme

```
ggplot(data=YourData, aes( x = xaxis, y = yaxis))+  
  geom_point() +  
  scale_x_continuous(...)+  
  theme(...)
```

# ggplot basics

- There's a sneaky thing...

```
ggplot(data=YourData, aes( x = xaxis, y = yaxis))+  
  geom_point() +  
  scale_x_continuous(...)+  
  theme(...)
```

# ggplot basics

- There's a sneaky thing...

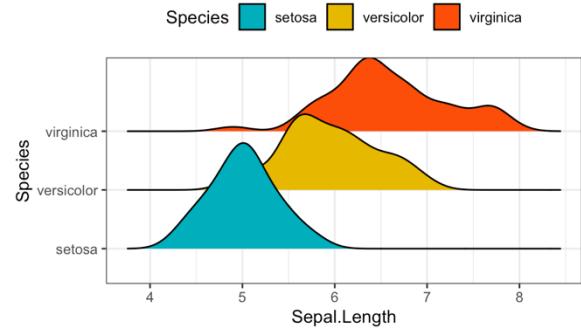
```
ggplot()+
  geom_point(data=YourData, aes( x = xaxis, y = yaxis))+
  scale_x_continuous(...)+
  theme(...)
```

# ggplot basics

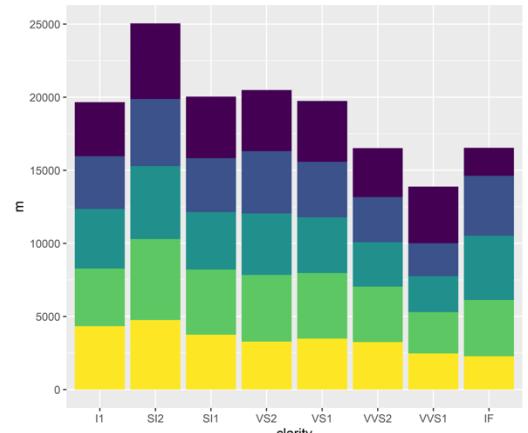
- Scale can apply to fills & colours too

```
ggplot()+
  geom_point(data=YourData, aes( x = xaxis, y = yaxis,
                                colour=ColourVariable))+
  scale_x_continuous(...)+
  scale_colour_manual(values=c(...))+  
  theme(...)
```

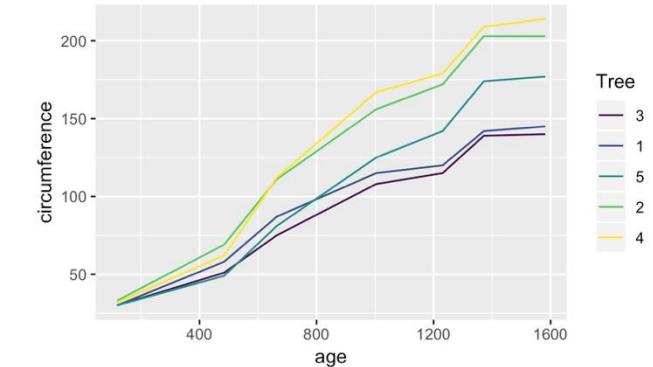
# What are some of the things you can do with ggplot?



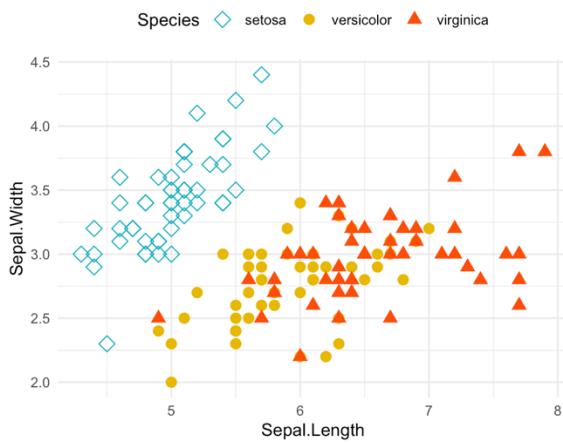
`geom_density(...)`



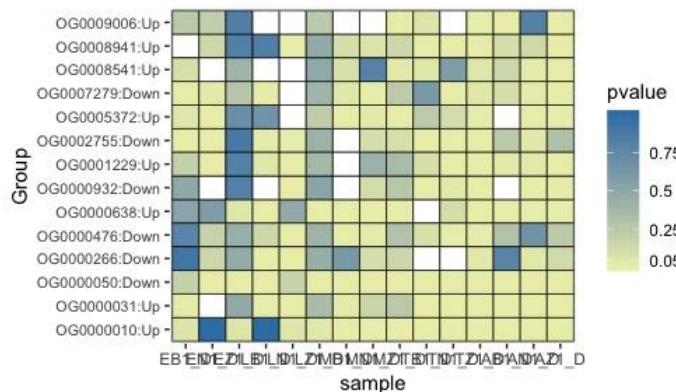
`geom_bar(position="stack", stat="identity")`



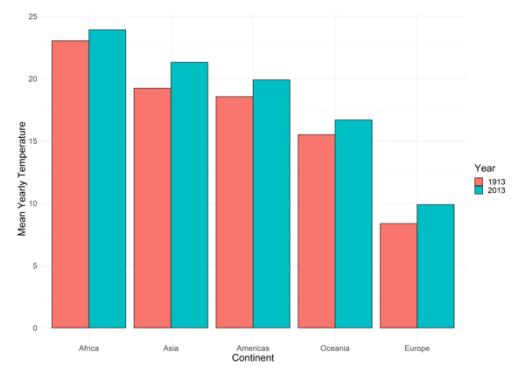
`geom_line(...)`



`geom_point(...)`



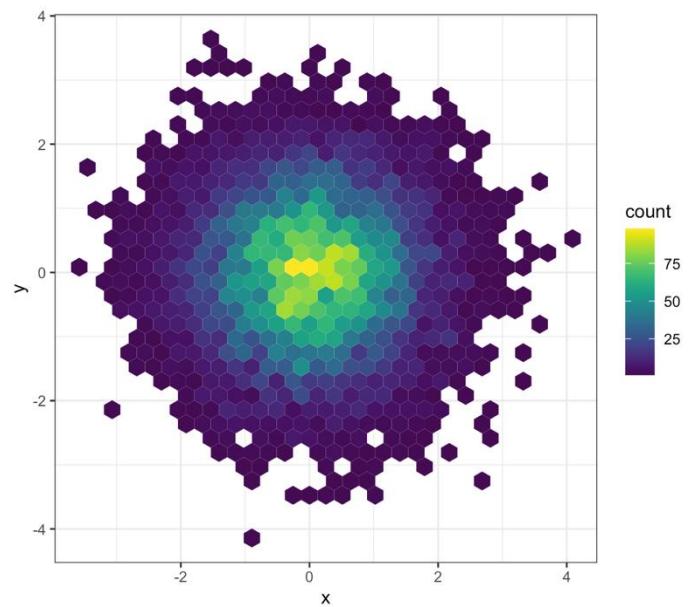
`geom_tile(...)`



`geom_bar(position="dodge", stat="identity")`

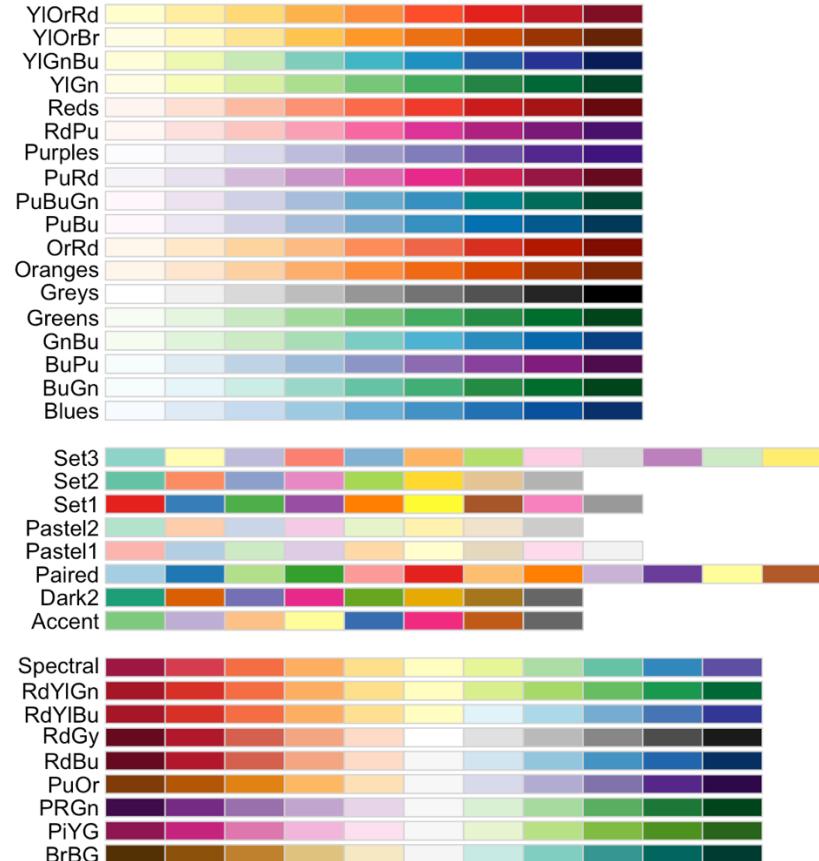
# Fun ggplot things to be aware of...

- The ‘viridis’ colour palette is great for colour blindness



# Fun ggplot things to be aware of...

- The ‘viridis’ colour palette is great for colour blindness
- The ‘RColorBrewer’ also has great colour combos (again esp good for colour blindness)



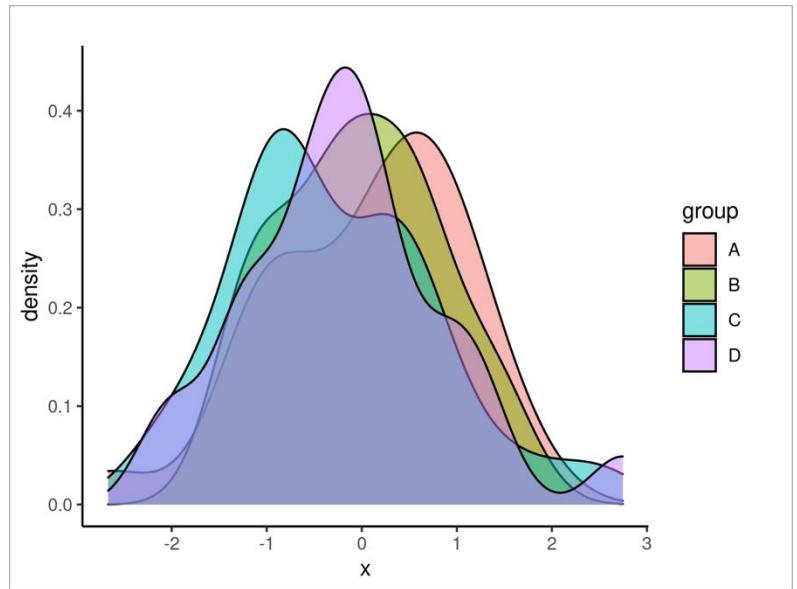
# Fun ggplot things to be aware of...

- The ‘viridis’ colour palette is great for colour blindness
- The ‘RColorBrewer’ also has great colour combos (again esp good for colour blindness)
- There’s an R package for Wes Anderson themed colour packages!



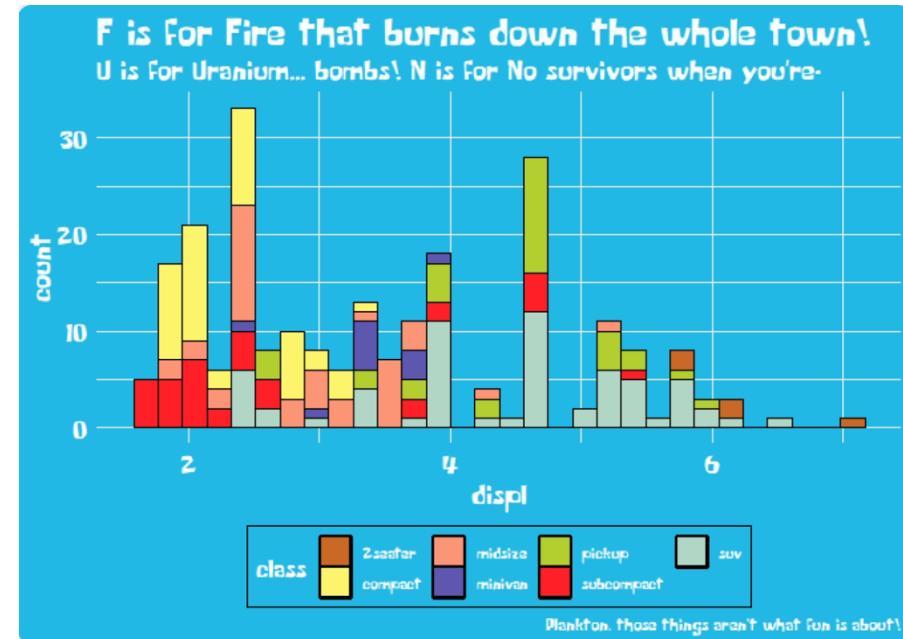
# Fun ggplot things to be aware of...

- The ‘viridis’ colour palette is great for colour blindness
- The ‘RColorBrewer’ also has great colour combos (again esp good for colour blindness)
- There’s an R package for Wes Anderson themed colour packages!
- There are a range of inbuilt themes to try:
  - I like `theme_classic(...)`



# Fun ggplot things to be aware of...

- The ‘viridis’ colour palette is great for colour blindness
- The ‘RColorBrewer’ also has great colour combos (again esp good for colour blindness)
- There’s an R package for Wes Anderson themed colour packages!
- There are a range of inbuilt themes to try:
  - I like `theme_classic(...)`
- See this excellent link for other themes!  
<https://rfortherestofus.com/2019/08/themes-to-improve-your-ggplot-figures>



# Data visualization with ggplot2 :: CHEATSHEET



## Basics

ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same components: a **data set**, a **coordinate system**, and **geoms**—visual marks that represent data points.



To display values, map variables in the data to visual properties of the geom (**aesthetics**) like **size**, **color**, and **x** and **y** locations.



Complete the template below to build a graph.

```
ggplot(data = <DATA>) +  
<GEOM_FUNCTION>(mapping = aes(<MAPPINGS>),  
stat = <STAT>, position = <POSITION>) +  
<COORDINATE_FUNCTION> +  
<FACET_FUNCTION> +  
<SCALE_FUNCTION> +  
<THEME_FUNCTION>
```

ggplot(data = mpg, aes(x = cty, y = hwy)) Begins a plot that you finish by adding layers to. Add one geom function per layer.

last\_plot() Returns the last plot.

ggsave("plot.png", width = 5, height = 5) Saves last plot as 5' x 5' file named "plot.png" in working directory. Matches file type to file extension.

**Aes** Common aesthetic values.  
**color** and **fill** - string ("red", "#RRGGBB")

**linetype** - integer or string (0 = "blank", 1 = "solid", 2 = "dashed", 3 = "dotted", 4 = "dotdash", 5 = "longdash", 6 = "twodash")

**size** - integer (in mm for size of points and text)

**linewidth** - integer (in mm for widths of lines)

**shape** - integer/shape name or a single character ("a")



## Geoms

Use a geom function to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

### GRAPHICAL PRIMITIVES

```
a <- ggplot(economics, aes(date, unemploy))  
b <- ggplot(seals, aes(x = long, y = lat))
```

**a + geom\_blank()** and **a + expand\_limits()**  
Ensure limits include values across all plots.

**b + geom\_curve(aes(yend = lat + 1,  
xend = long + 1), curvature = 1) -> x, yend, y, yend,  
alpha, angle, color, curvature, linetype, size**

**a + geom\_path(lineend = "butt",  
linejoin = "round", linemetre = 1)**  
x, y, alpha, color, group, linetype, size

**a + geom\_polygon(aes(alpha = 50)) -> x, y, alpha,  
color, fill, group, subgroup, linetype, size**

**b + geom\_rect(aes(xmin = long, ymin = lat,  
xmax = long + 1, ymax = lat + 1)) -> xmax, xmin,  
ymax, ymin, alpha, color, fill, linetype, size**

**a + geom\_ribbon(aes(ymin = unemploy - 900,  
ymax = unemploy + 900)) -> x, y, ymin, alpha,  
color, fill, group, linetype, size**

### LINE SEGMENTS

common aesthetics: x, y, alpha, color, linetype, size

**b + geom\_abline(aes(intercept = 0, slope = 1))**  
**b + geom\_hline(aes(intercept = lat))**  
**b + geom\_vline(aes(xintercept = long))**

**b + geom\_segment(aes(yend = lat + 1, xend = long + 1))**  
**b + geom\_spoke(aes(angle = 1:1155, radius = 1))**

### ONE VARIABLE continuous

```
c <- ggplot(mpg, aes(hwy)); c2 <- ggplot(mpg)
```

**c + geom\_area(stat = "bin")**  
x, y, alpha, color, fill, linetype, size

**c + geom\_density(kernel = "gaussian")**  
x, y, alpha, color, fill, group, linetype, size, weight

**c + geom\_dotplot()**  
x, y, alpha, color, fill

**c + geom\_freqpoly()**  
x, y, alpha, color, group, linetype, size, weight

**c + geom\_histogram(binwidth = 5)**  
x, y, alpha, color, fill, linetype, size, weight

**c2 + geom\_qq(aes(sample = hwy))**  
x, y, alpha, color, fill, linetype, size, weight

### discrete

```
d <- ggplot(mpg, aes(fl))
```

**d + geom\_bar()**  
x, alpha, color, fill, linetype, size, weight

### TWO VARIABLES both continuous

```
e <- ggplot(mpg, aes(cty, hwy))
```

**e + geom\_label(aes(label = cty), nudge\_x = 1,  
nudge\_y = 1) -> x, y, label, alpha, angle, color,  
family, fontface, hjust, lineheight, size, vjust**

**e + geom\_point()**  
x, y, alpha, color, fill, shape, size, stroke

**e + geom\_quantile()**  
x, y, alpha, color, group, linetype, size, weight

**e + geom\_rug(sides = "bl")**  
x, y, alpha, color, fill, linetype, size

**e + geom\_smooth(method = lm)**  
x, y, alpha, color, fill, group, linetype, size, weight

**e + geom\_text(aes(label = cty), nudge\_x = 1,  
nudge\_y = 1) -> x, y, label, alpha, angle, color,  
family, fontface, hjust, lineheight, size, vjust**

### one discrete, one continuous

```
f <- ggplot(mpg, aes(class, hwy))
```

**f + geom\_col()**  
x, y, alpha, color, fill, group, linetype, size

**f + geom\_boxplot()**  
x, y, lower, middle, upper, ymax, ymin, alpha,  
color, fill, group, linetype, shape, size, weight

**f + geom\_dotplot(binaxis = "y", stackdir = "center")**  
x, y, alpha, color, fill, group

**f + geom\_violin(scale = "area")**  
x, y, alpha, color, fill, group, linetype, size, weight

### both discrete

```
g <- ggplot(diamonds, aes(cut, color))
```

**g + geom\_count()**  
x, y, alpha, color, fill, shape, size, stroke

**e + geom\_jitter(height = 2, width = 2)**  
x, y, alpha, color, fill, shape, size

### THREE VARIABLES

```
sealsSz <- with(seals, sqrt(delta_long^2 + delta_lat^2)); l <- ggplot(seals, aes(long, lat))
```

**l + geom\_contour(aes(z = z))**  
x, y, z, alpha, color, group, linetype, size, weight

**l + geom\_raster(aes(fill = z), hjust = 0.5,  
vjust = 0.5, interpolate = FALSE)**  
x, y, alpha, fill

**l + geom\_contour\_filled(aes(fill = z))**  
x, y, alpha, color, fill, group, linetype, size, subgroup

**l + geom\_tile(aes(fill = z))**  
x, y, alpha, color, fill, linetype, size, width

### continuous bivariate distribution

```
h <- ggplot(diamonds, aes(carat, price))
```

**h + geom\_bin2d(binwidth = c(0.25, 500))**  
x, y, alpha, color, fill, linetype, size, weight

**h + geom\_density\_2d()**  
x, y, alpha, color, group, linetype, size

**h + geom\_hex()**  
x, y, alpha, color, fill, size

### continuous function

```
i <- ggplot(economics, aes(date, unemploy))
```

**i + geom\_area()**  
x, y, alpha, color, fill, linetype, size

**i + geom\_line()**  
x, y, alpha, color, group, linetype, size

**i + geom\_step(direction = "hv")**  
x, y, alpha, color, group, linetype, size

### visualizing error

```
df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2)  
j <- ggplot(df, aes(grp, fit, ymin = fit - se, ymax = fit + se))
```

**j + geom\_crossbar(fatten = 2) -> x, y, ymax,  
ymin, alpha, color, fill, group, linetype, size**

**j + geom\_errorbar() -> x, ymax, ymin,  
alpha, color, group, linetype, size, width**  
Also **geom\_errorbarh()**.

**j + geom\_linerange()**  
x, ymin, ymax, alpha, color, group, linetype, size

**j + geom\_pointrange() -> x, y, ymin, ymax,  
alpha, color, fill, group, linetype, shape, size**

### maps

Draw the appropriate geometric object depending on the simple features present in the data. aes() arguments:  
map\_id, alpha, color, fill, linetype, linewidth.

```
nc <- sf::st_read(system.file("shape/nc.shp", package = "sf"))
```

**ggplot(nc) +  
geom\_sf(aes(fill = AREA))**

# Break

# Activity 1

Read the Instructions in the Code  
(Slowly, Carefully!)

**STOP AT LINE 73 AND GET YOUR  
PENS OUT!!**

Submit your plot!

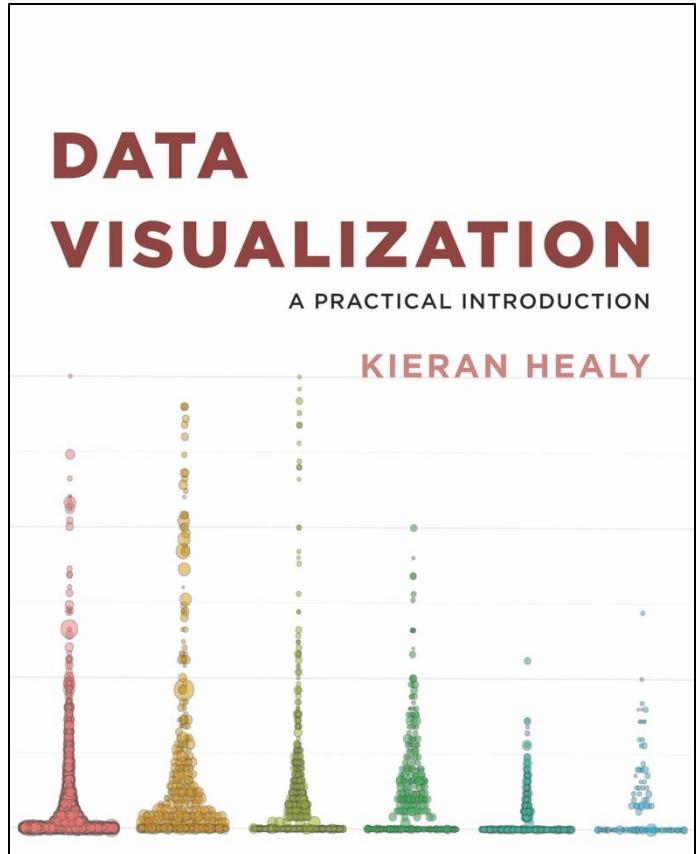
[hwauchop@ed.ac.uk](mailto:hwauchop@ed.ac.uk)

Label with “yourname\_ugly”

(or pretty, depending on what you went for!)

Plots due by 3:10!!

# Impress My Nieces!!



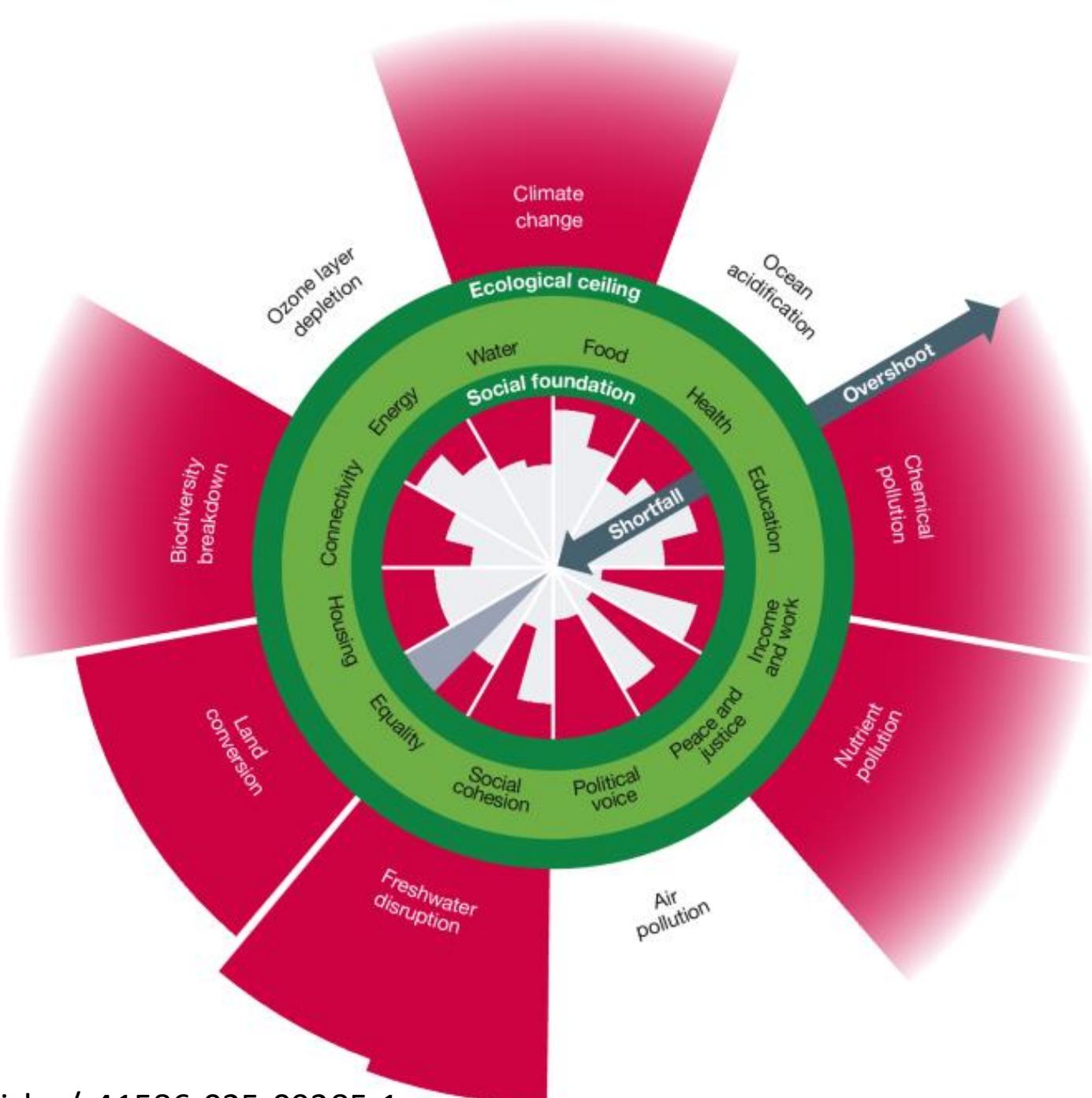
Winner wins this book!!

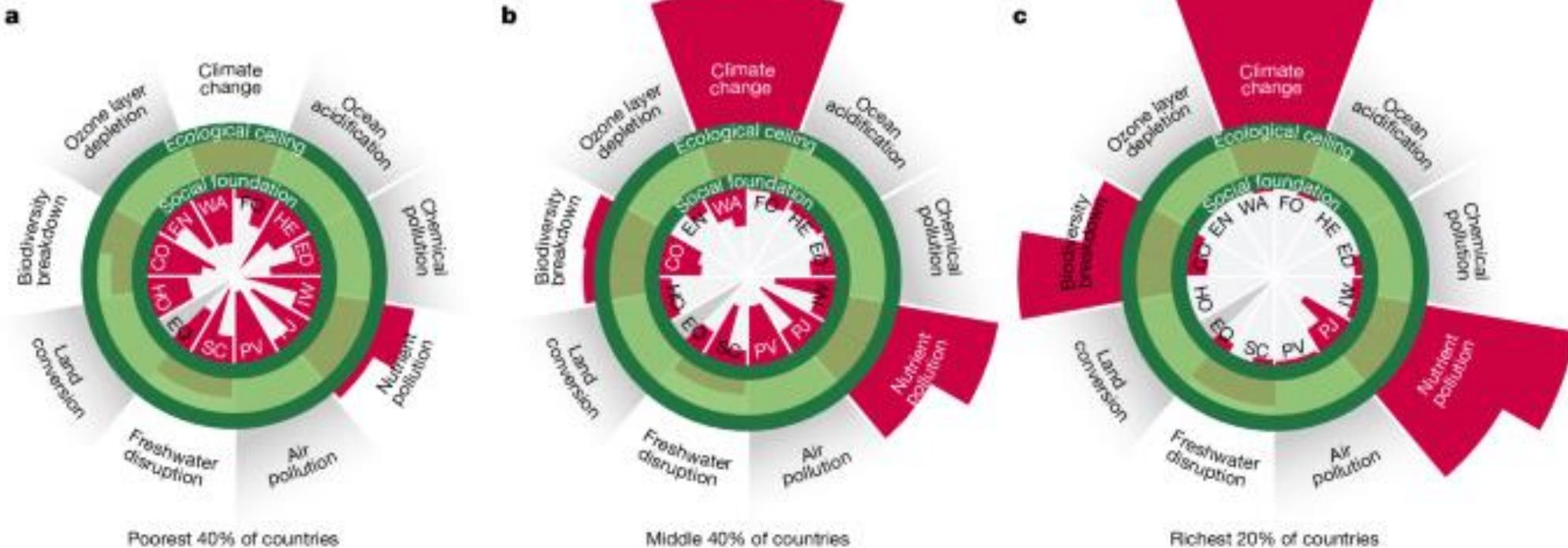
Submit by Friday 24<sup>th</sup> Oct

The girls must be able to understand and describe the result

The girls like:

- Animals esp lions and shinglebag skinks
- Their dog, who loves cuddles beaches and catching crabs
- They love travel and have been to 21 countries
- Plus art, icecream singing
- They love pastel, rainbow colours esp turquoise, blue and yellow





FO	Food
HE	Health
ED	Education
IW	Income and work

WA	Water
EN	Energy
CO	Connectivity
HO	Housing

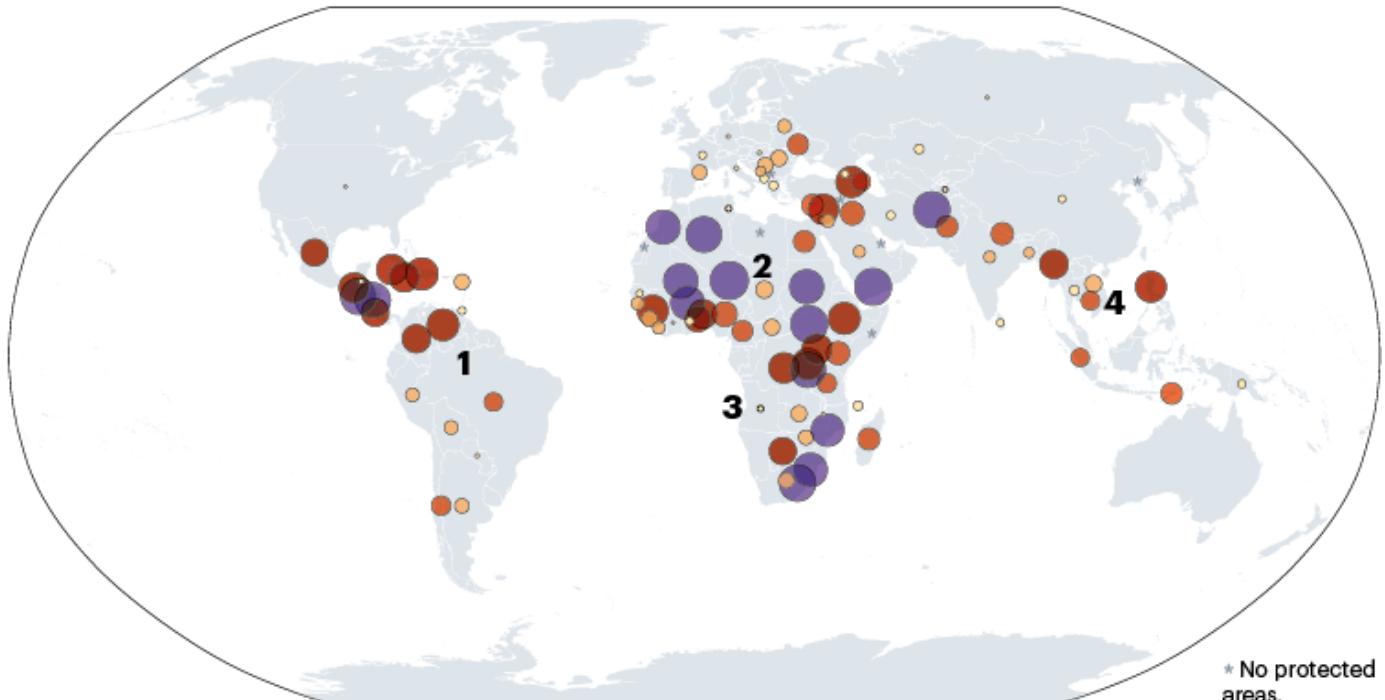
EQ	Equality
SC	Social cohesion
PV	Political voice
PJ	Peace and justice

## FAR-REACHING EFFECTS

Today, 39 countries, including many that are rich in biodiversity, are experiencing sustained or escalating levels of conflict.

Proportion of protected areas affected by armed violence (%;  
1 January 2019 to  
31 December 2023)

- > 70
- 40–70
- 20–40
- 5–20
- < 5



**1** In Brazil and Venezuela, armed violence is mainly associated with organized crime.

**2** In Niger, conflicts affect 95% of protected land.

**3** Some wildlife populations in remote areas of Angola have been able to persist because people moved to towns and cities to avoid conflict and landmines.

**4** In Cambodia, conflict has helped an illegal wildlife trade to emerge by increasing people's access to weapons.

