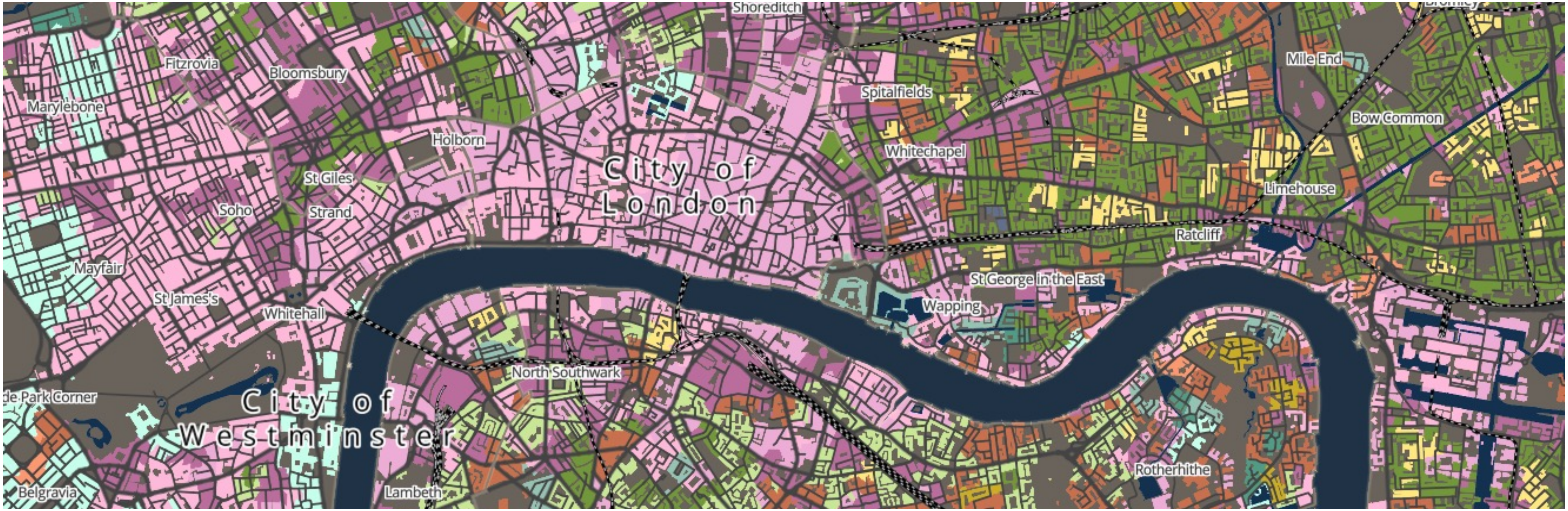
Geocomputation

Programming for Spatial Analysis



UCL

Where are we at?

Part I: Foundational Concepts

W1	Geocomputation: An Introduction	
W2	GIScience and GIS software	QGIS
W3	Cartography and Visualisation	
W4	Programming for Data Analysis	R
W5	**Programming for Spatial Analysis**	

This week

- Data science principles	Context
- Using R as a GIS	
- Managing data with the `tidyverse`	Data management
- Managing spatial data with `sf`	
- Visualising spatial data with `tmap` and `ggplot2`	Visualising spatial data
- Some practical notes on coding	Miscellaneous

Context

Data Science Principles

Repeatability: the same methodology will produce the same (or nearly the same) outputs given the same inputs and reduces opportunity for error.

Reproducibility: work can be easily redone/completed by another (i.e. they have all information needed).

Data Science Principles

Collaboration: easy to share work with others and collaborate, preferably in real- time, with others, alongside easy integration with version control.

Scalability: basic – can re-run work easily, adjusting variables and parameters to include additional data; intermediate – can expand on work to include larger datasets; advanced – suitable for distributed computing.

Programming languages

“Everyone does need to learn to code. It is no longer sufficient for a GI Scientists to just work with a standard GIS interface: menus, buttons and black boxes.”

Brunsdon and Comber 2020

Using R as a GIS

- R is our programming language; RStudio is our Integrated Development Environment to develop and run R code.
- RStudio is now a high functionality piece of software – and can be used in many ways like a traditional GUI statistical software.
- However: there are a lot of differences compared to using a traditional GUI GIS software.

Using R as a GIS

- No map canvas – we do not ”see” our spatial data when it is loaded.
- When we load spatial data, it is loaded into the memory as a variable – we have to actively plot it using the base `plot()` function or a more advanced visualisation library to see our data.
- We can see the attribute table of our vector data through the `View()` function – this will load as a table.

Using R as a GIS

- When we use spatial analysis tools in QGIS, we often create new data files in the process – or we actively export new data files to save our edits. In R, we use variables in our processing and analysis.
- As a result, our analysis results and outputs are stored as variables. We need to actively export our outputs if we want to:
  - Share them or use within a different programme
  - Avoid re-running our scripts each time we want to use the outputs

Using R as a GIS

- No map composer / print layout to create our final maps.
- We do have visualisation libraries to create maps – that can then be automatically saved into PNGs or PDFs.
- Learning curve, but it is less fiddly than QGIS’ Print Layout – and if you spot errors in your processing/analysis, updating your maps becomes a simple case of re-running your code.

When not to use R as a GIS?

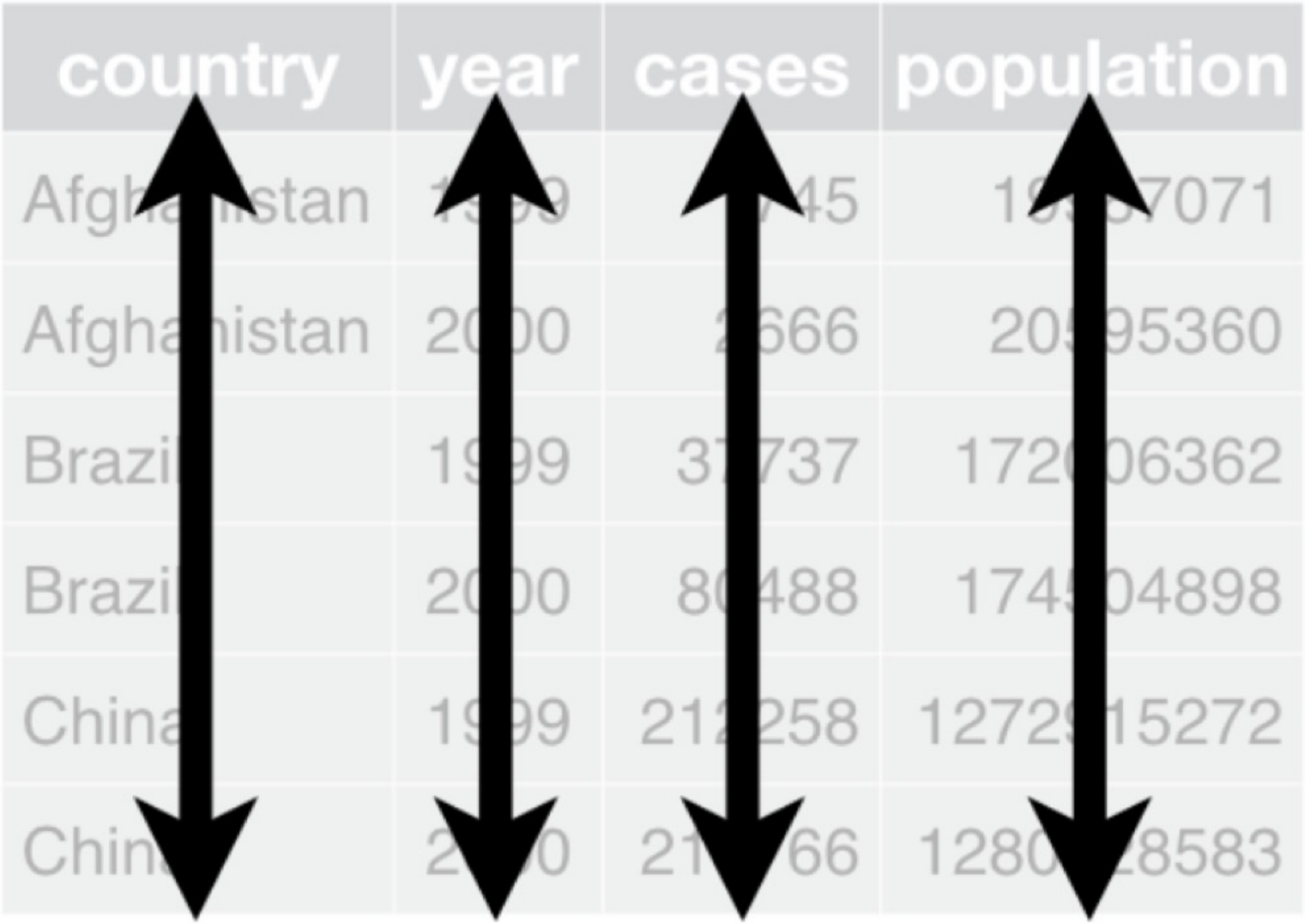
- When digitising data (not covered in this module) or editing the geometry of spatial data (e.g. needing to move boundaries, change location of points).
- For fine-tuning cartographic outputs sometimes only a desktop GUI GIS will do.
- Interactive data exploration (although possible with interactive views).

Managing data

Managing data

- Wickham 2014. 80 percent of your time goes to data cleaning and preparation (‘data wrangling’).
- Tidy data refers to the structure and organisation of your data set.
- The idea boils down to three principles.
- Brought together in the tidyverse.

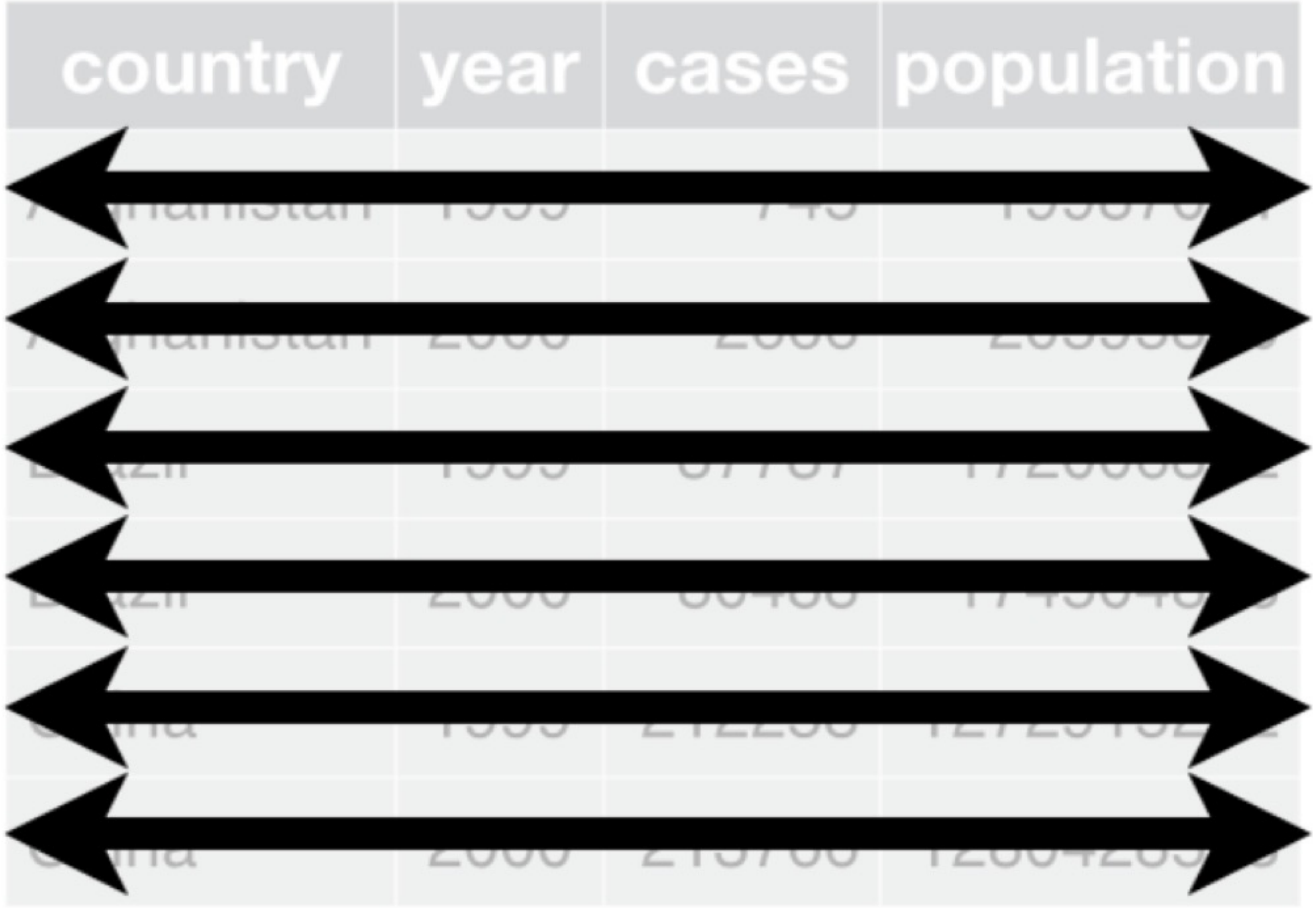
Tidy data



country	year	cases	population
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	213766	1280428583

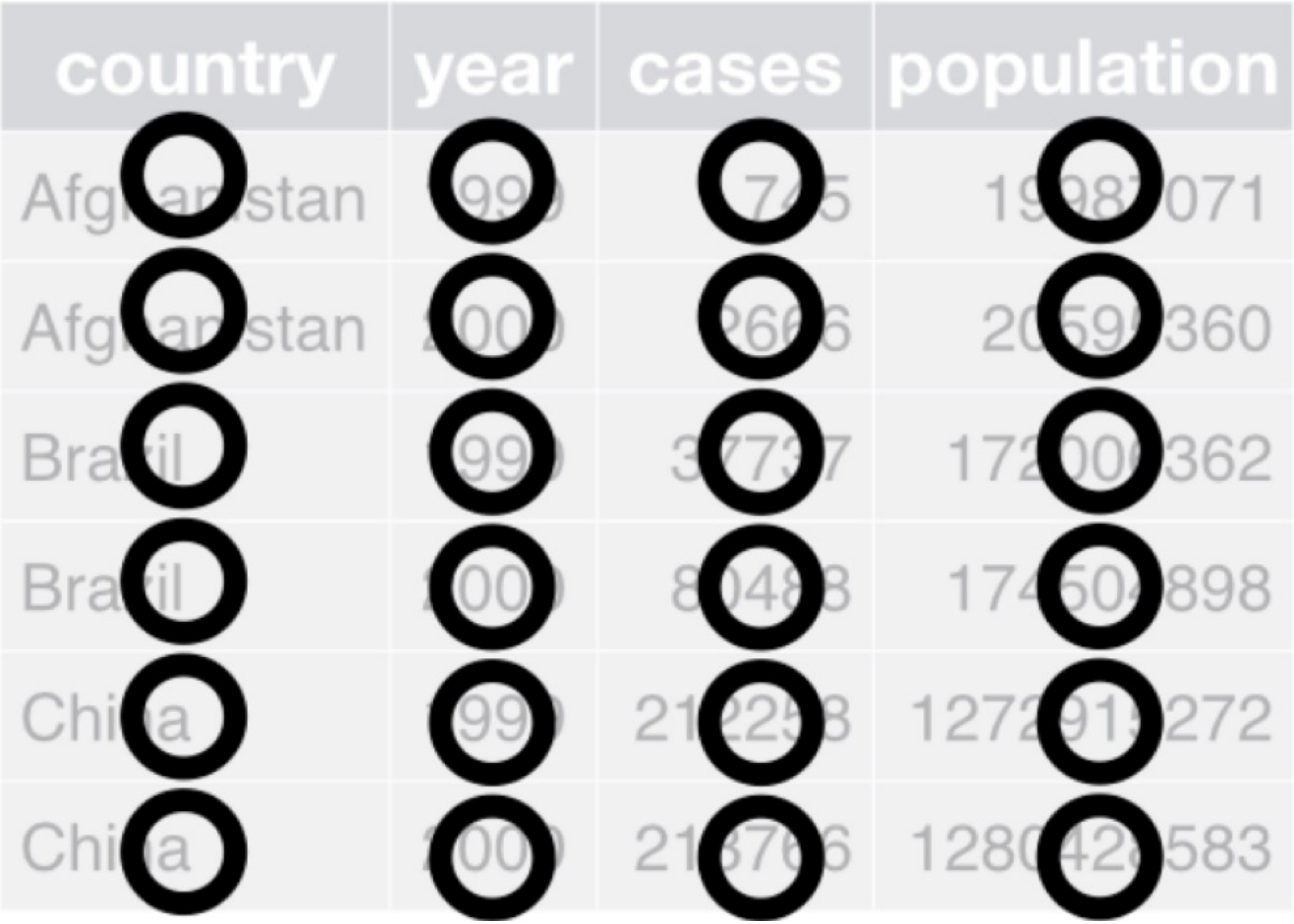
Each variable must have its own column

Tidy data



Each observation must have its own row

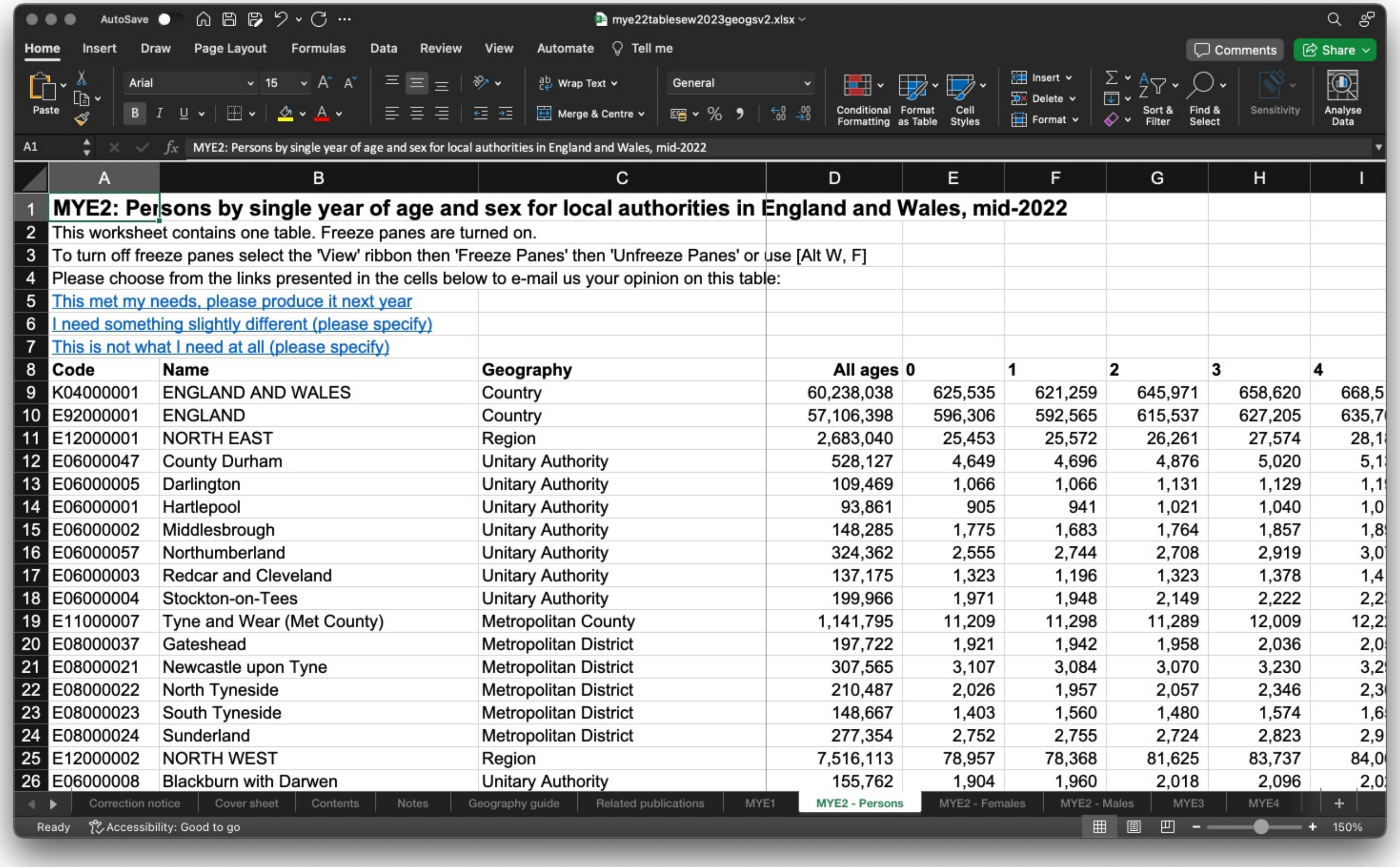
Tidy data



country	year	cases	population
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	213766	1280428583

Each value must have its own cell

Tidy data



MYE2: Persons by single year of age and sex for local authorities in England and Wales, mid-2022

This worksheet contains one table. Freeze panes are turned on.

To turn off freeze panes select the 'View' ribbon then 'Freeze Panes' then 'Unfreeze Panes' or use [Alt W, F]

Please choose from the links presented in the cells below to e-mail us your opinion on this table:

This met my needs, please produce it next year

I need something slightly different (please specify)

This is not what I need at all (please specify)

Code	Name	Geography	All ages	0	1	2	3	4
K04000001	ENGLAND AND WALES	Country	60,238,038	625,535	621,259	645,971	658,620	668,5
E92000001	ENGLAND	Country	57,106,398	596,306	592,565	615,537	627,205	635,7
E12000001	NORTH EAST	Region	2,683,040	25,453	25,572	26,261	27,574	28,1
E06000047	County Durham	Unitary Authority	528,127	4,649	4,696	4,876	5,020	5,1
E06000005	Darlington	Unitary Authority	109,469	1,066	1,066	1,131	1,129	1,1
E06000001	Hartlepool	Unitary Authority	93,861	905	941	1,021	1,040	1,0
E06000002	Middlesbrough	Unitary Authority	148,285	1,775	1,683	1,764	1,857	1,8
E06000057	Northumberland	Unitary Authority	324,362	2,555	2,744	2,708	2,919	3,0
E06000003	Redcar and Cleveland	Unitary Authority	137,175	1,323	1,196	1,323	1,378	1,4
E06000004	Stockton-on-Tees	Unitary Authority	199,966	1,971	1,948	2,149	2,222	2,2
E11000007	Tyne and Wear (Met County)	Metropolitan County	1,141,795	11,209	11,298	11,289	12,009	12,2
E08000037	Gateshead	Metropolitan District	197,722	1,921	1,942	1,958	2,036	2,0
E08000021	Newcastle upon Tyne	Metropolitan District	307,565	3,107	3,084	3,070	3,230	3,2
E08000022	North Tyneside	Metropolitan District	210,487	2,026	1,957	2,057	2,346	2,3
E08000023	South Tyneside	Metropolitan District	148,667	1,403	1,560	1,480	1,574	1,6
E08000024	Sunderland	Metropolitan District	277,354	2,752	2,755	2,724	2,823	2,9
E12000002	NORTH WEST	Region	7,516,113	78,957	78,368	81,625	83,737	84,0
E06000008	Blackburn with Darwen	Unitary Authority	155,762	1,904	1,960	2,018	2,096	2,0

Common errors

- Column headers are values rather than variable names.
- Multiple variables are stored in one column.
- Variables are stored in both rows and columns.
- Multiple observational units are stored in the same column.
- A single observation is stored in multiple tables.

Tidy ?

country	year	type	count
Afghanistan	2019	cases	745
Afghanistan	2019	population	19 987 071
Afghanistan	2020	cases	2 666
Afghanistan	2020	population	20 595 360
Brazil	2019	cases	3,7737
Brazil	2019	population	172 006 362
Brazil	2020	cases	80 488
Brazil	2020	population	174 504 898
China	2019	cases	212 258
China	2019	population	1 272 915 272
China	2020	cases	213 766
China	2020	population	1 280 428 583

Tidy ?

country	year	rate
Afghanistan	2019	745 / 19,987,071
Afghanistan	2020	2,666 / 20,595,360
Brazil	2019	3,7737 / 172,006,362
Brazil	2020	80,488 / 174,504,898
China	2019	212,258 / 1,272,915,272
China	2020	213,766 / 1,280,428,583

Tidy ?

Cases

country	**2019**	**2020**
Afghanistan	745	2 666
Brazil	3,7737	80 488
China	212 258	213 766

Population

country	**2019**	**2020**
Afghanistan	19 987 071	20 595 360
Brazil	172 006 362	174 504 898
China	1 272 915 272	1 280 428 583

Tidy ?

country	year	cases	population
Afghanistan	2019	745	19 987 071
Afghanistan	2020	2 666	20 595 360
Brazil	2019	3,7737	172 006 362
Brazil	2020	80 488	174 504 898
China	2019	212 258	1 272 915 272
China	2020	213 766	1 280 428 583

Managing spatial data

- R has the capacity to read, load and store a range of file formats.
- Functions in both the base R library plus a huge host of software-specific packages (e.g. STATA, SPSS) for reading, writing and converting data between different file formats associated with those specific software (e.g. from a SPSS file to a `csv` etc.)
- Base R does not handle the reading, loading, and storing of spatial data.

Managing spatial data

- How do we read in spatial data?
- GDAL: Geospatial Data Abstraction Library (*reading, writing*)
- GEOS: Geometry Engine Open Source (*spatial operations*)

Managing spatial data

`sp`

- ‘Classes and methods for spatial data’
- First development in using spatial data in R (2005)
- Not fully compliant with the dataframe format

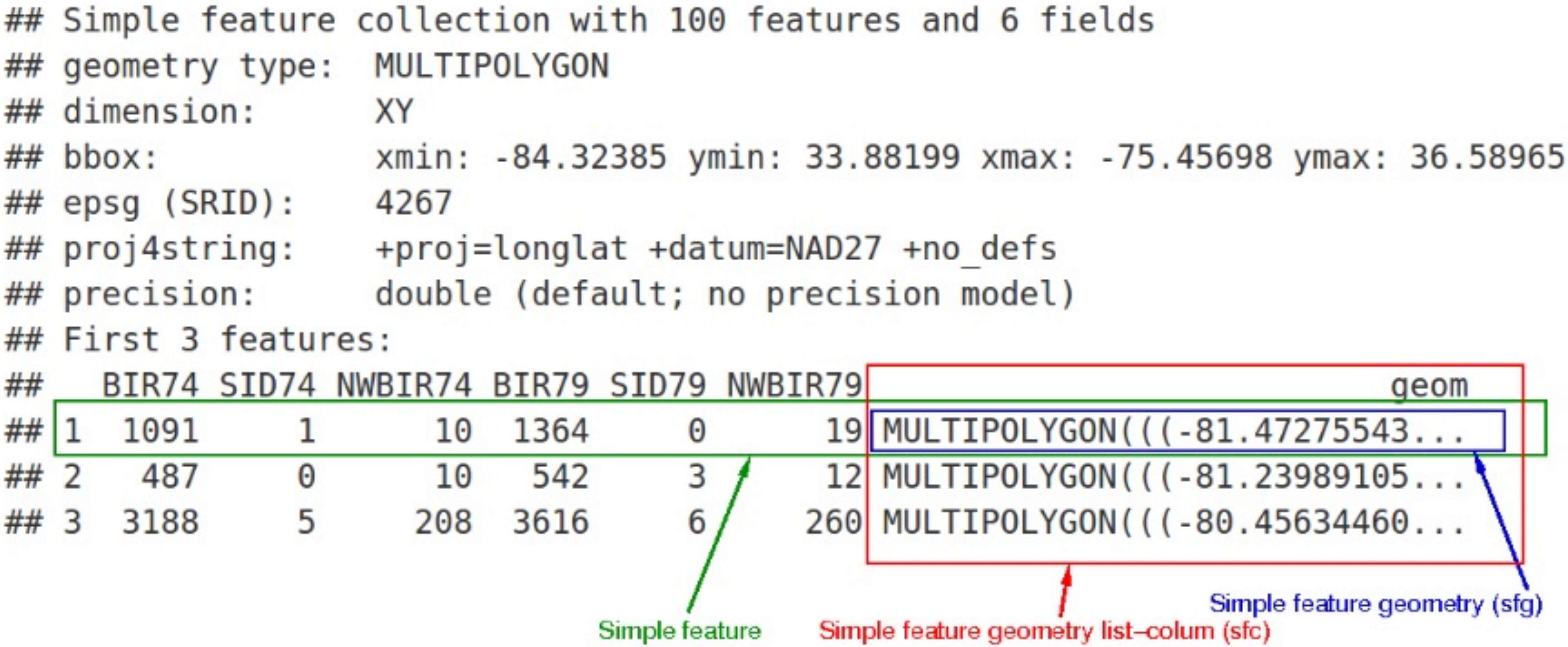
`sf`

- ‘Support for simple features, a standardized way to encode spatial vector data’
- Fully compliant with the dataframe format

Managing spatial data

- The `sf` (simple features) package facilitates the storage, access and management of geometric objects stored as simple features in R.
- Importantly: sf objects are dataframes with a geometry column, containing WKT geometries at the end.

Managing spatial data



```
## Simple feature collection with 100 features and 6 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: -84.32385 ymin: 33.88199 xmax: -75.45698 ymax: 36.58965
## epsg (SRID):    4267
## proj4string:    +proj=longlat +datum=NAD27 +no_defs
## precision:      double (default; no precision model)
## First 3 features:
##   BIR74 SID74 NWBIR74 BIR79 SID79 NWBIR79                          geom
## 1  1091     1      10  1364     0      19 MULTIPOLYGON(((-81.47275543...
## 2   487     0      10   542     3      12 MULTIPOLYGON(((-81.23989105...
## 3  3188     5     208  3616     6     260 MULTIPOLYGON(((-80.45634460...
```

Simple feature geometry (sfg)
Simple feature
Simple feature geometry list–colum (sfc)

Managing spatial data

The `sf` package contains a huge set of tools for:

- Reading and writing spatial data.
- Querying a range of different point, line and polygon vector geometries
- `st_` prefix for all functions identical to that used in PostGIS (SQL) queries
- `st_` originates from SQL association of “Spatial Temporal”.

RStudio

● LIVE

Visualising spatial data

Libraries for spatial data visualisation

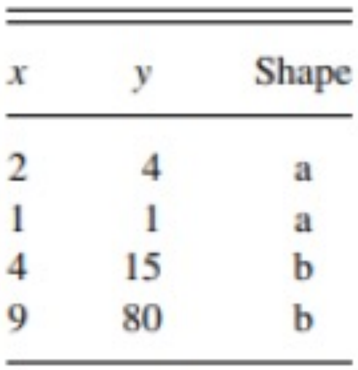
- A huge variety of packages that facilitate visualisation of spatial data.
- Most common: `tmap` and `ggplot2`
- Both are based on the “Layered Grammar of Graphics”

Grammar of Graphics

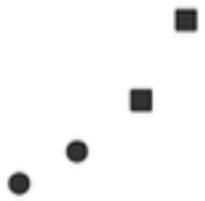
- Main concept: graphics can be built up through multiple layers of data.
- Values in a dataset are examples of aesthetics – values that can be viewed in a graphic.
- Data, scales and coordinate systems and plot annotations can then be layered on top of these data values to produce the final graphic.

Grammar of Graphics

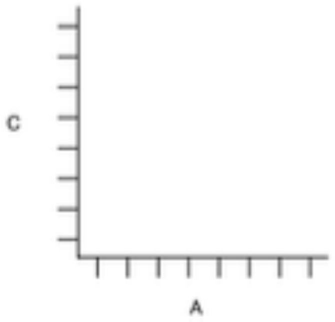
Dataframe



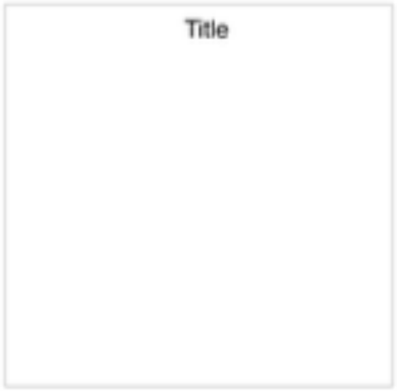
x	*y*	Shape
2	4	a
1	1	a
4	15	b
9	80	b



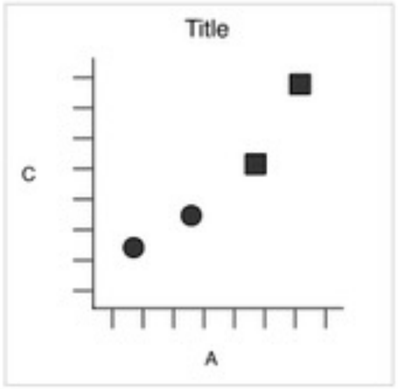
Dataframe values



Dataframe scale



Dataframe annotations



Final Graphic

RStudio

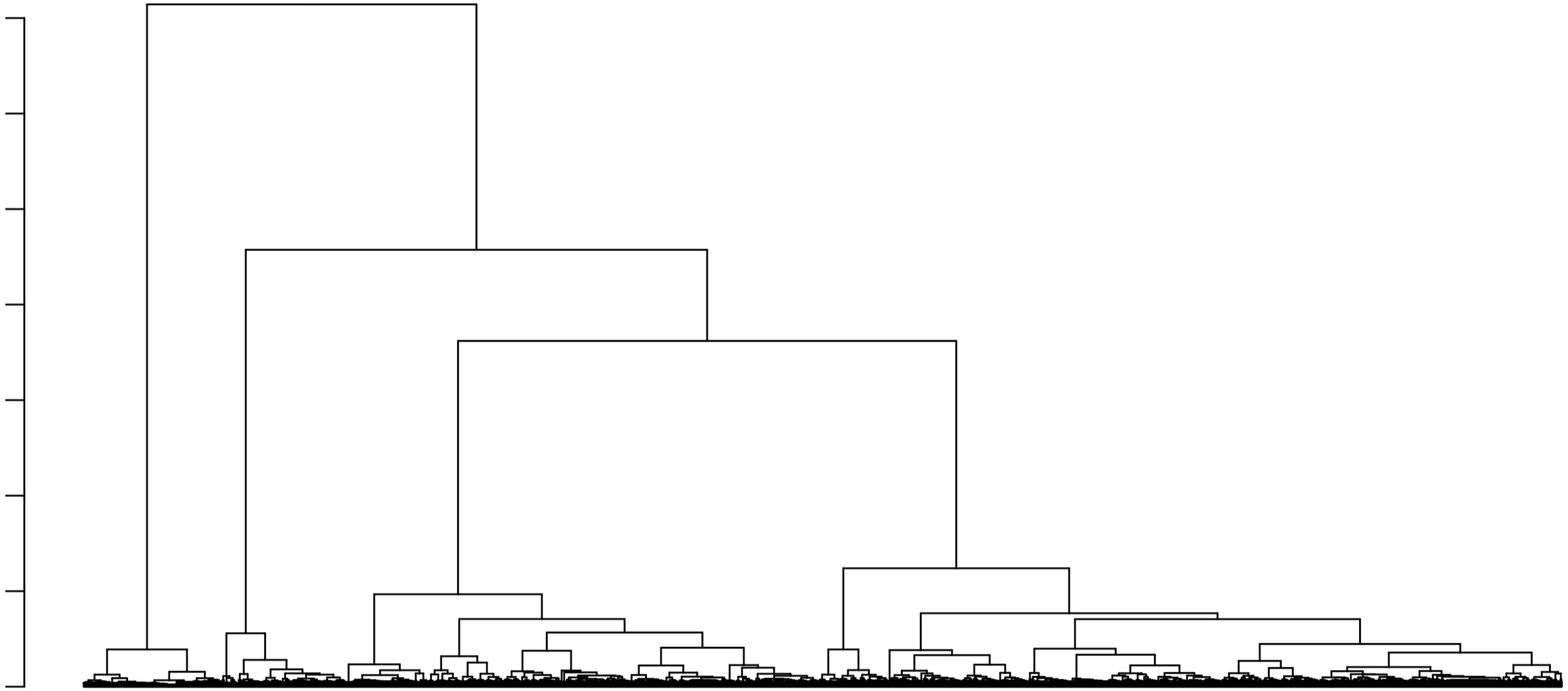
● LIVE

Miscellaneous

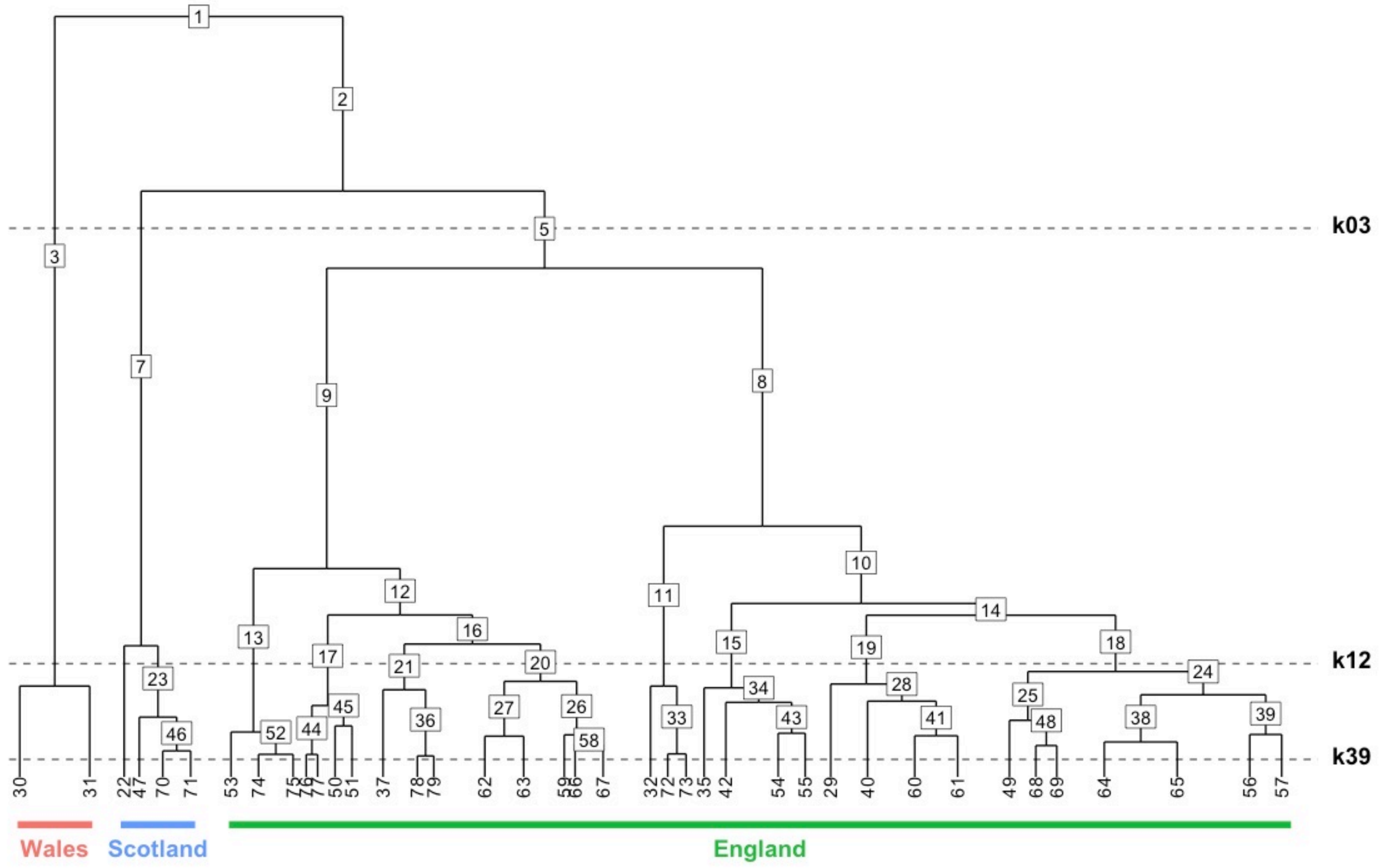
Comment your code

- Ensures that you remember what you did, especially with long pieces of code.
- Helps others to follow what you have done.
- In a way: in-text metadata

Comment your code



Comment your code



Version control

- The practice of tracking and managing changes to software code.
- Example: Git (GitHub, GitLab).
- “Track changes" on steroids.
- Beyond the scope of this module but you can still set up your own small version control system (`”_v0.1`”, “`v_0.2`”, “`v_1.0`”).

RStudio

● LIVE

Conclusion

- In many ways Geospatial / Geographic Information Systems have evolved considerably since their invention in the 1970s.
- Analysis has become more sophisticated. Processes are very similar, but the scale and size of datasets we are using require more computing resources.
- R was developed as a piece of statistical software, but being open source, coupled with a few key players making early developments in spatial analysis and visualisation means it is now one of the most comprehensive GIS solutions available today.

Map challenge

Have a good reading week

Questions

Justin van Dijk

j.t.vandijk@ucl.ac.uk