HDAT9800 Visualisation and Communication of Health Data

Chapter 1 - Parts 1 & 2

Dr Tim Churches & Dr James Farrow

HDAT9800 Visualisation and Communication of Health Data

Chapter 1 - Part 1

James Farrow & Tim Churches

Welcome to HDAT9800

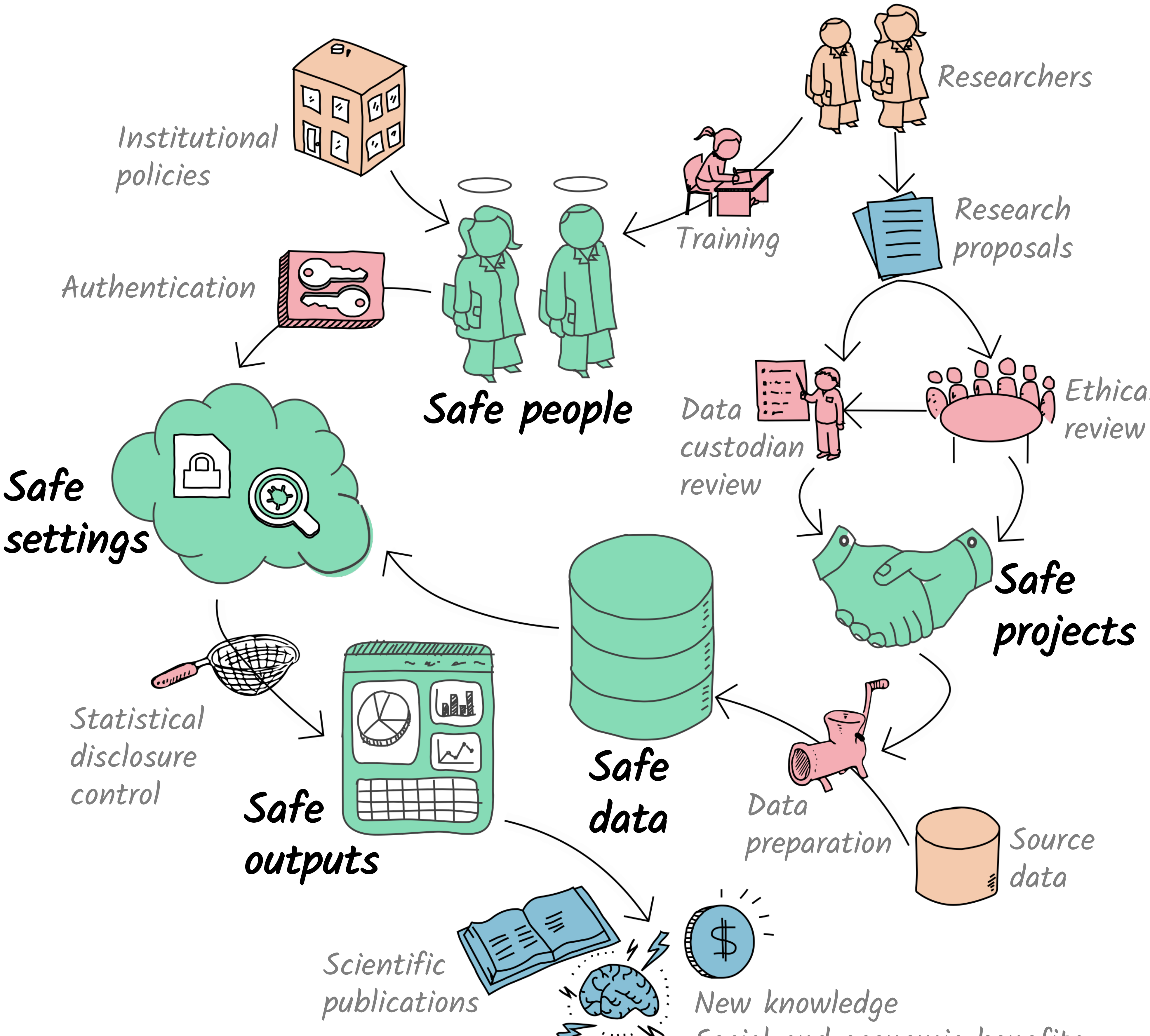
Course aims (foundational)

- learn some best practice methods for data manipulation and data management which aren't covered in other MSc HDS courses
- learn and practice best practice in reproducible research
    - data analyses which you, or others, cannot easily reproduce at a later date on a different computer are bad data analyses
- learn and use best practice for collaborating on health data science work
    - in real life (cf doing course work), you will rarely be working alone

Welcome to HDAT9800

Course aims

- learn useful techniques (backed by a little theory) for visualisation of health data and analyses
    - in R
    - but applicable to many other software ecosystems, which we will mention from time to time: python, SAS, Stata, D3.js
- we will cover statistical maps, but we don't cover geographical information systems (GIS)
- focus is on statistical charts and graphics, not *infographics*


Institutional policies
Researchers
Training
Research proposals
Authentication
Safe people
Data custodian review
Ethical review
Safe settings
Safe projects
Safe data
Statistical disclosure control
Safe outputs
Data preparation
Source data
Scientific publications
New knowledge
Social and economic benefits

Welcome to HDAT9800

Course aims, part 2

- learn useful techniques (backed by a little theory) for communication of health data analyses
    - emphasis is on written and online communication
        - reports, scientific papers, blogs and web sites, dashboards and apps
    - we will cover creating slide decks for oral presentations, but this course does not aim to make you a better public speaker (there are a lot of very good, more general courses which cover that)
        - see also https://www.nature.com/articles/d41586-021-01281-8

Foundation course creator/ presenter

Dr James Farrow

- Computer Scientist: background in physics and computer science
- Interests in data modelling, graph data approaches, data linkage, natural language processing, machine learning, geocoding and spatial visualisation

Foundation course creator/co-convener/presenter

Dr Tim Churches

- background in clinical medicine, particularly general practice and geriatrics, in public health medicine, in population health informatics and in clinical epidemiology.
- spent nearly two decades in NSW Health building key public health information and population health surveillance systems.
- most recently he has been working on COVID-19 monitoring and simulation models (which we will examine in detail as extension material in this course) and application of machine learning to clinical cancer data management
- contact Tim through Teams or email timothy.churches@unsw.edu.au

Course assistant

Dami Sotade

- contact Dami through Teams or email o.sotade@unsw.edu.au

Course format

Weekly exposition (lecture) component

- 60 to 120 minutes per week, pre-recorded lectures or talks like this
- accompanying slide decks and/or learn tutorials

Weekly interactive session

- 90 to 120 minutes per week, via Zoom videoconferencing
- Q&A on and discussion of lecture material
- additional curriculum material and/or hands-on enrichment

Course format

Assessment

- Weekly assignments on core material (for seven weekly chapters): 7x8% → 56% of total marks
  - these are due three weeks after the beginning of each weekly chapter
- Two additional assignments requiring individual research and initiative: 2x7% → 14% of total marks
- Group assignment: 30% of total marks

Contact procedures

You may contact the instructors through the Teams space for the course, of via email: timothy.churches@unsw.edu.au

We undertake to respond as soon as possible, almost always the same day (except for weekends — we will monitor Teams and emails for help requests from students on weekends, but we may not be able to respond until the following Monday).

We may not always be able to solve your problem or answer your question immediately, but we will at least acknowledge it and start working on it.

Administrative problems

Administrative problems (including requests for special consideration) should be sent to Tim Churches at timothy.churches@unsw.edu.au

Course thematic outline

- Introduction: *dplyr* / *git*
- Literate programming for presentations and publications: *knitr*
- Basic charting: *ggplot2*
- Collaboration: *git*
- Advanced visualisation, including mapping: *leaflet,* various *ggplot2* extensions, visualisation of categorical data with *vcd,* visualisation of high-dimensional data using dimensional reduction methods *(PCA, t-SNE [t-distributed stochastic neighbour embedding])*

Enrichment topics

- Not necessarily in this order
- Using back-end databases for Big Data: *dbplyr*
- Advanced data manipulation/functional programming: *purrr*
- Reproducible scientific papers: *rticle* / *knitr*
- Reproducible presentations: *xaringan* / *knitr*
- Blogging (in which you create your own blog site, which you can keep using after the course): *distill* / *knitr*

Enrichment topics, continued

- Embedding charts in tables and text: *sparklines, gt*
- Interactive visualisations: *D3.js* via *htmlwidgets*
- 3D charts and maps: *rayshader / rayrender*

The point is communication

We want to be effective communicators

What is more effective?

A table of numbers?

A chart?

A map?

It’s a trick question: all of them can be useful

A Brief History of Data Visualisation

The following slides are closely based on a wonderful book chapter by Michael Friendly, a psychologist and statistician at York University in Canada. Recommended reading if you are interested in the history of data visualisation.

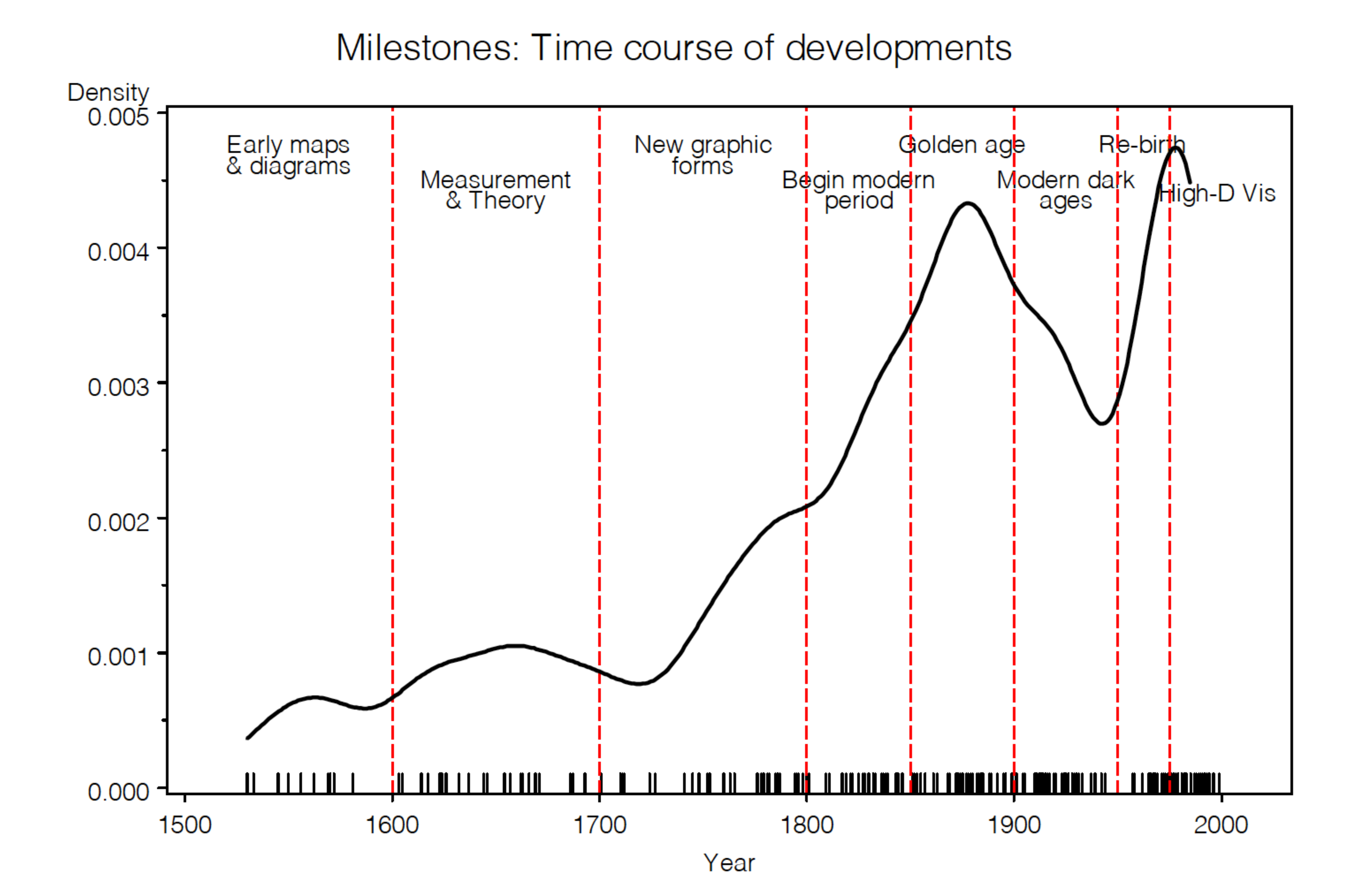
Friendly M. *A Brief History of Data Visualisation*. In: Chen C, Härdles W, Unwin A. (eds). *Handbook of Computational Statistics: Data Visualisation*. pp15-56. Springer-Verlag, Heidelberg, 2006. DOI:10.1007/978-3-540-33037-0_2

See also the Milestones Project at https://www.datavis.ca/milestones for more examples from the history of data visualisation

Data visualisation is not new

- it is often assumed that statistical graphics and data visualisation is a late 20th Century development (using computers)
    - if fact, data visualisation dates back to the 16th Century, and statistical graphics were largely developed in the 19th Century
- we won't be covering the history of cartography (maps) here, it has its own rich history, too extensive to discuss here

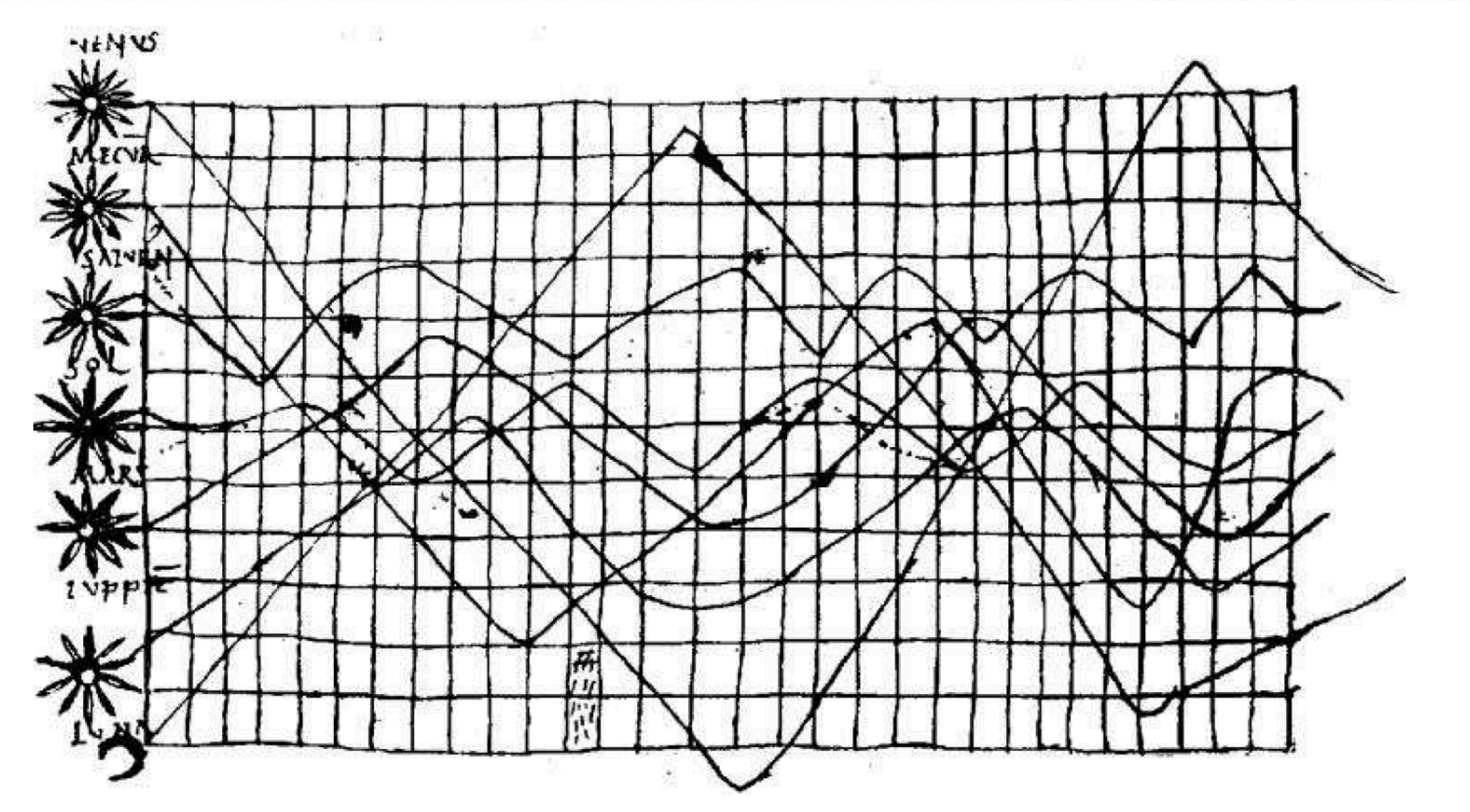
Data visualisation is not new



Source: Friendly M. *A Brief History of Data Visualisation. op.cit.*

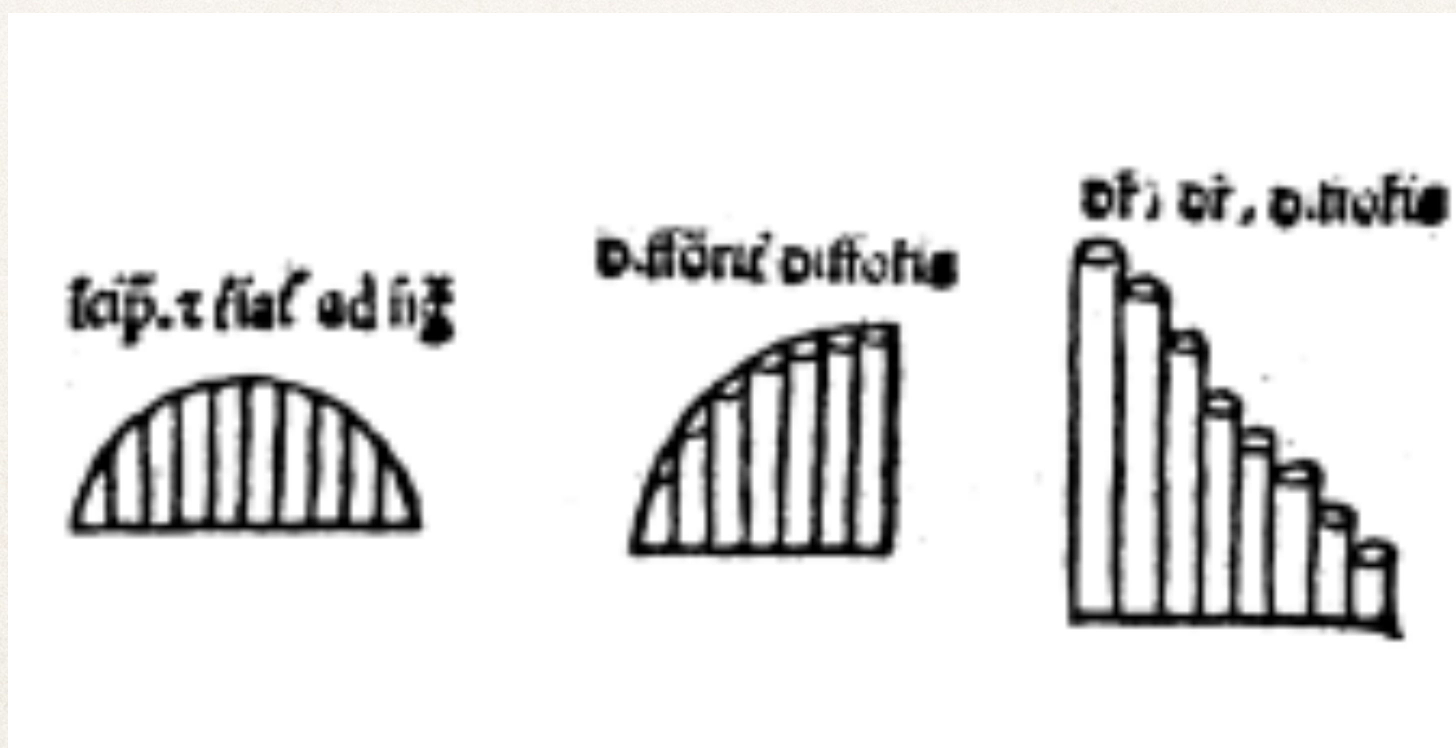
Pre-17th Century

- earliest known visualisation of quantitative information dates from the 10th Century, by an anonymous author, showing the positions of the seven most prominent heavenly bodies in space and time — it is a time-series chart



Pre-17th Century

- in the 14th Century, the idea of plotting a theoretical mathematical function as a proto-bar chart was introduced by Nicole Oresme, Bishop of Liseus (below), followed several decades later by a theoretical graph of distance versus speed by Nicholas of Cusa.

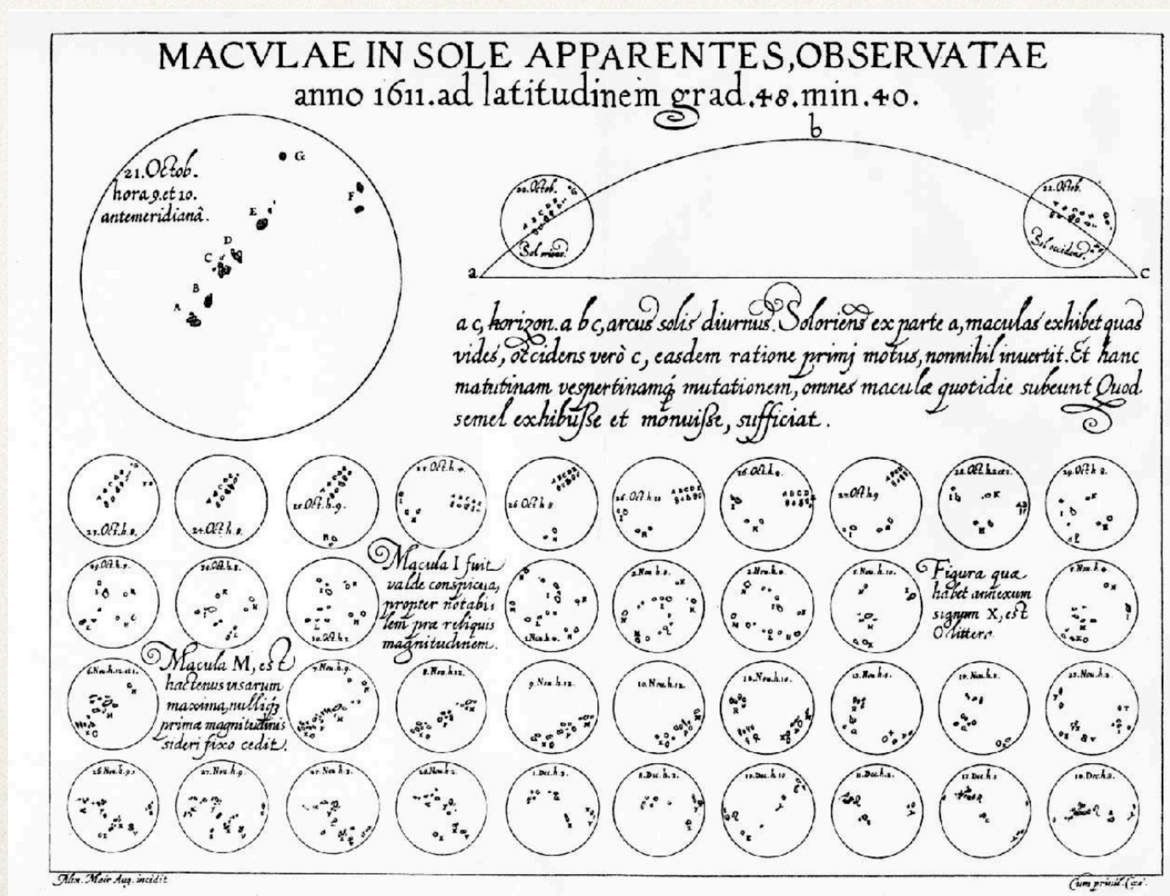


17th Century: measurement theory & early statistics

- physical measurement was a key concern, driven by astronomy, map making and navigation
- analytic geometry and coordinate systems were introduced by Descartes, theories of errors of measurement and estimation were developed by Galileo, Pascal and Fermat laid the foundations of probability theory, John Graunt founded the science of demographics, and William Petty introduced “political arithmetic”, which laid the foundations for modern economics

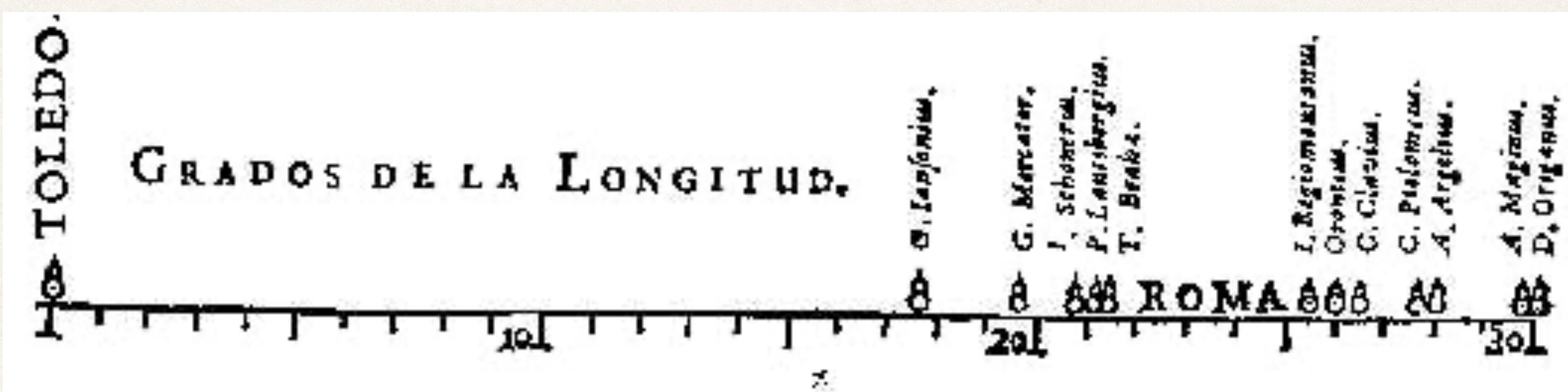
17th Century: Cristopher Sneer and "small multiples"

- Sneer introduced what Edward Tufte would much later call "small multiples" to show changes in sun spots over time
  - we will see how "small multiples" has recently become an important principle in statistical graphics eg "facets" in *ggplot2* in R



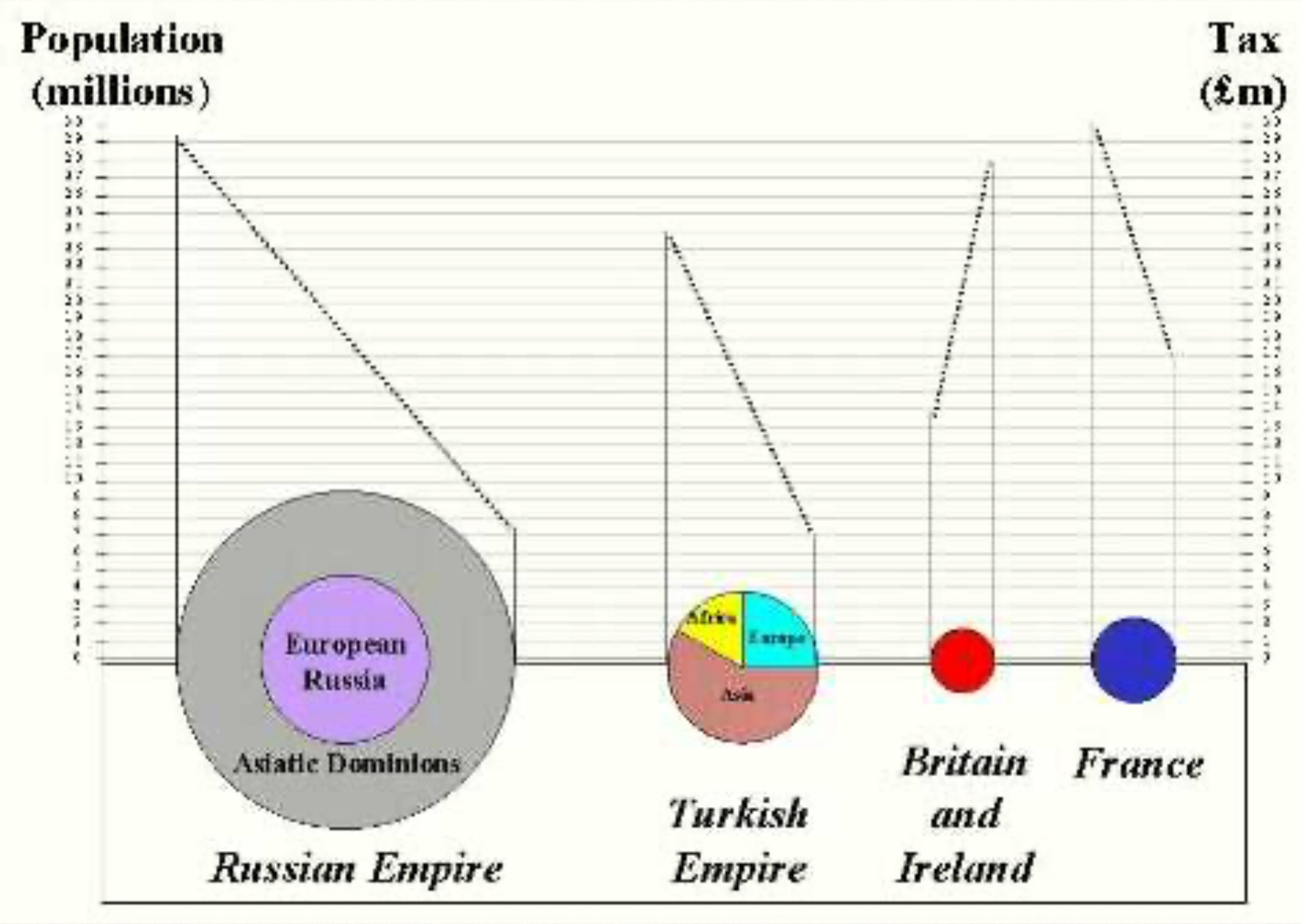
17th Century: the first statistical graph

- in 1644, Langren, a Flemish astronomer to the court of Spain, represented the 12 known estimates of the the longitude of Toledo relative to Rome (the correct value is 16 degree 30 minutes)



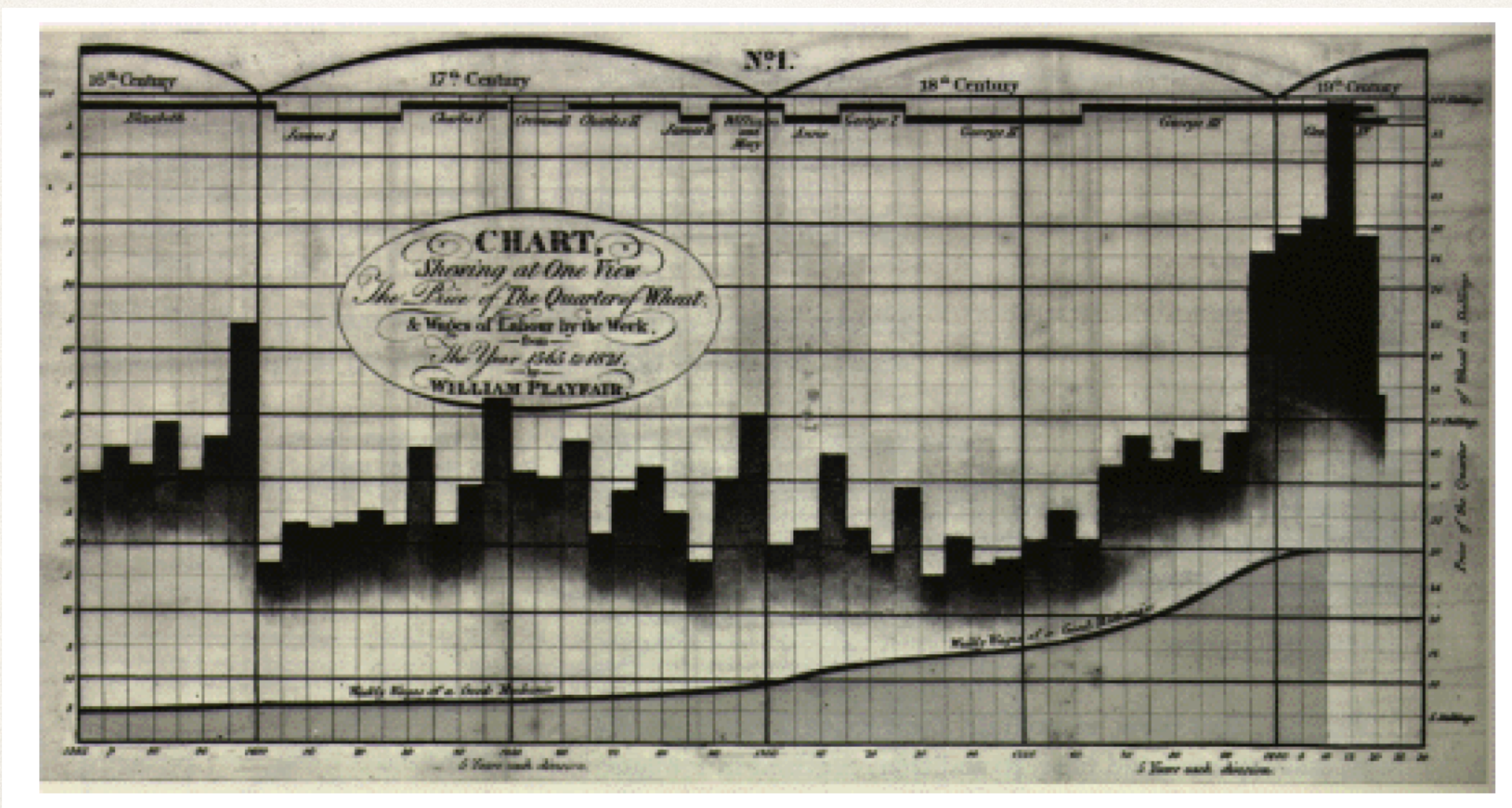
18th Century: foundations of statistical graphs

- William Playfair (1759-1823) invented most of the basic data graphs we use today, including line charts, bar charts, pie charts and circle graphs (one of his charts from 1801 is shown below)



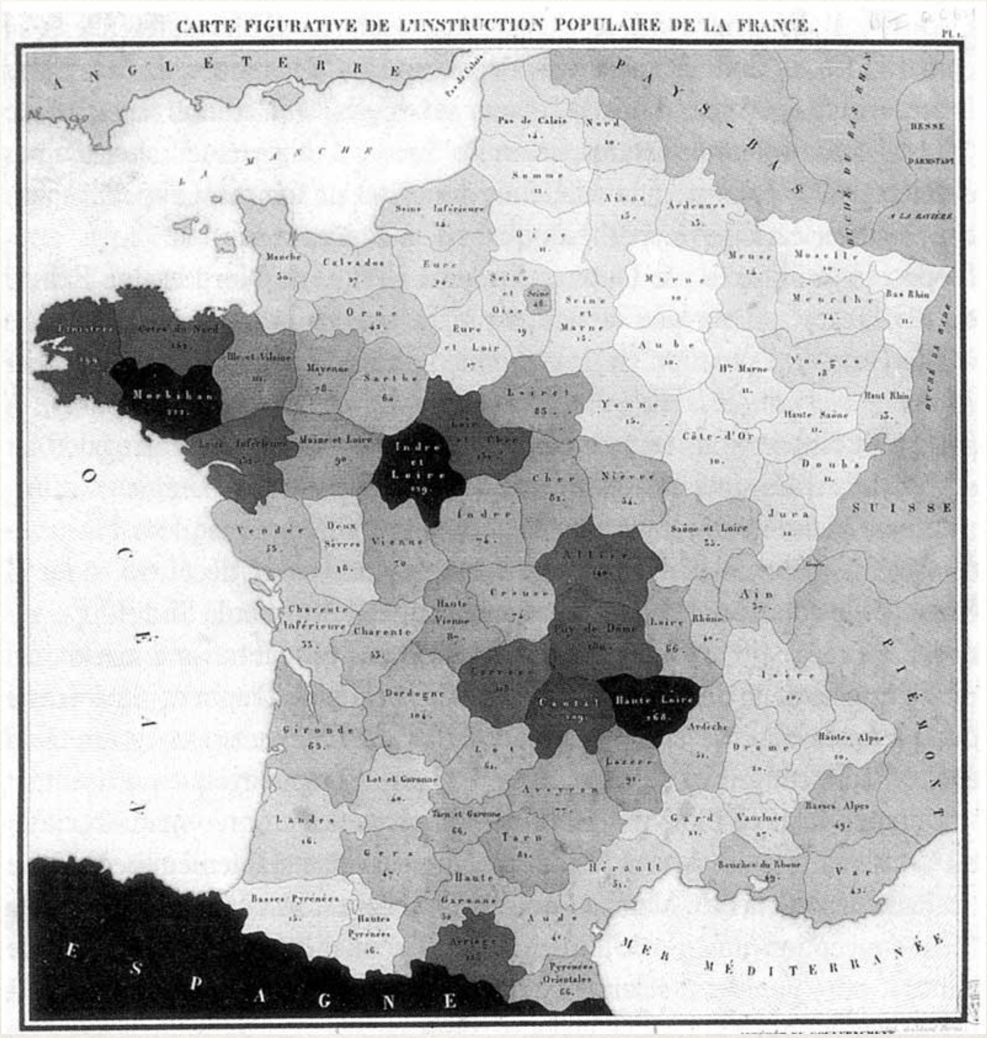
18th Century: foundations of statistical graphs

- William Playfair’s most famous chart, from 1826, showing time series of prices, wages and ruling monarch over a 250 year period



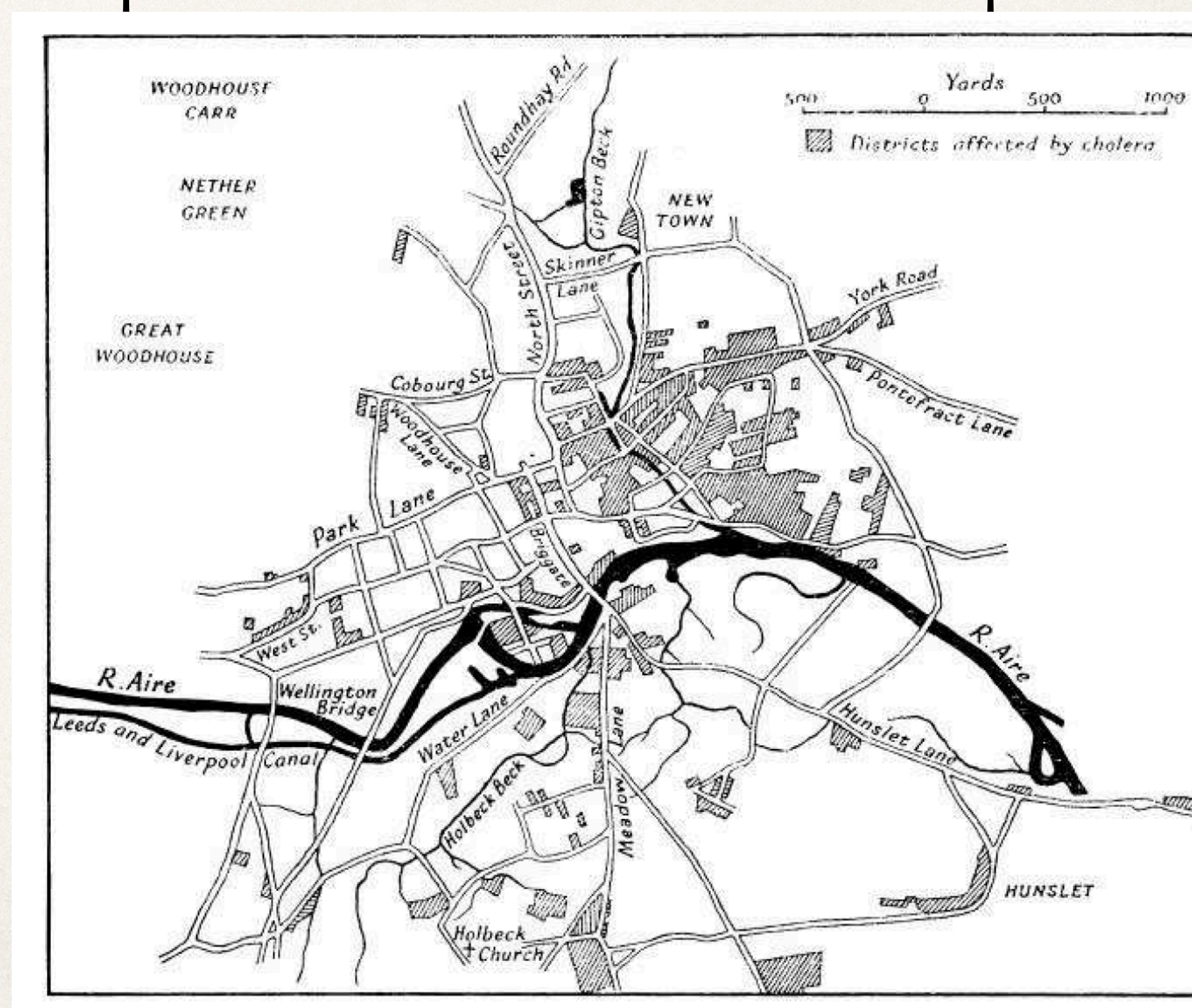
19th Century: origins of statistical maps

- In 1826, Baron Charles Dupin used continuous shadings (from black to white) to show illiteracy by province in France — this is the first example of a choropleth thematic statistical map



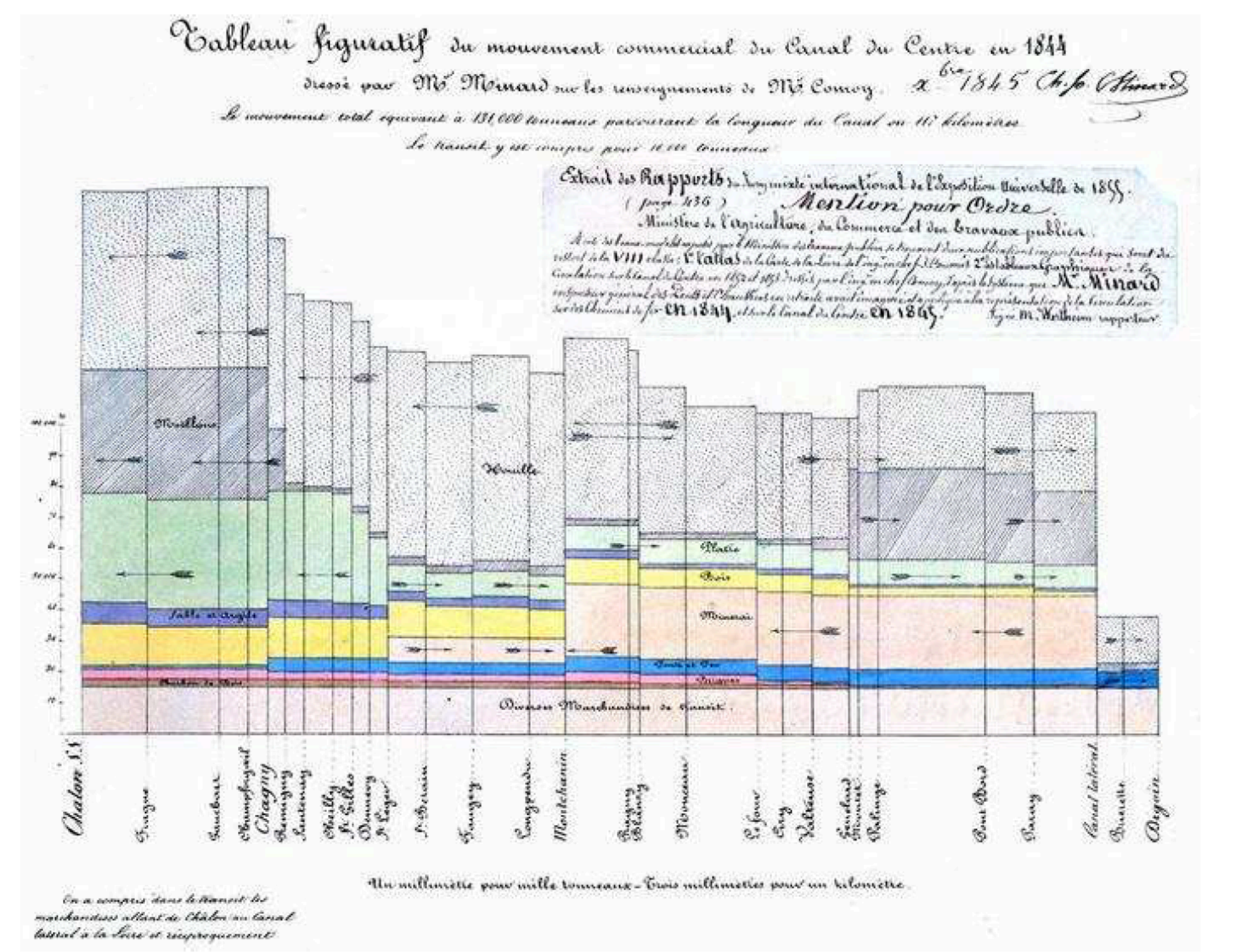
19th Century: epidemiological maps

- John Snow's 1855 map of cholera cases near the Broad St water pump in London (below left) is rightly famous in public health, but Robert Baker's 1833 map of cholera cases in Leeds predates it

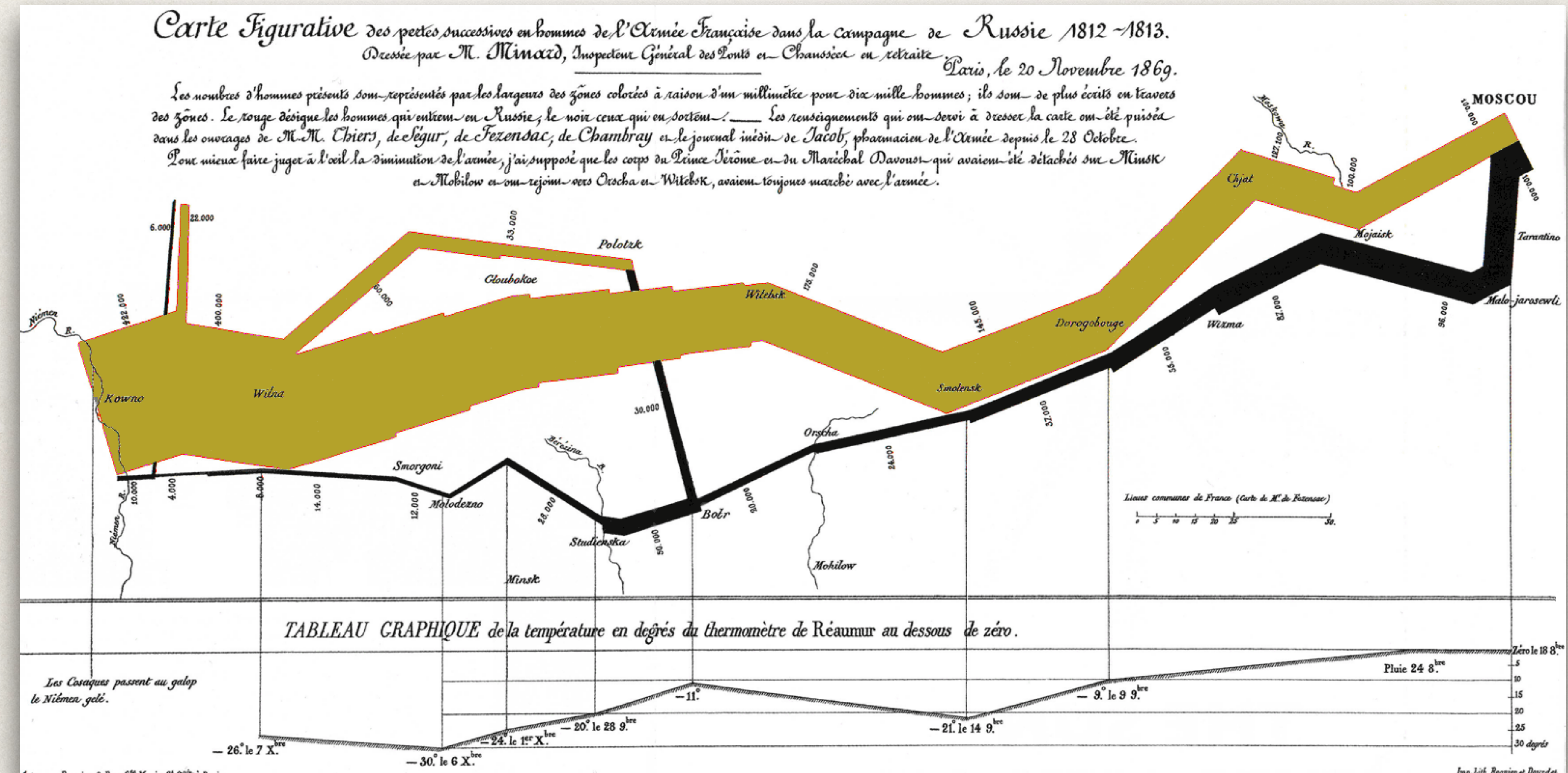


19th Century: *Tableau graphique*

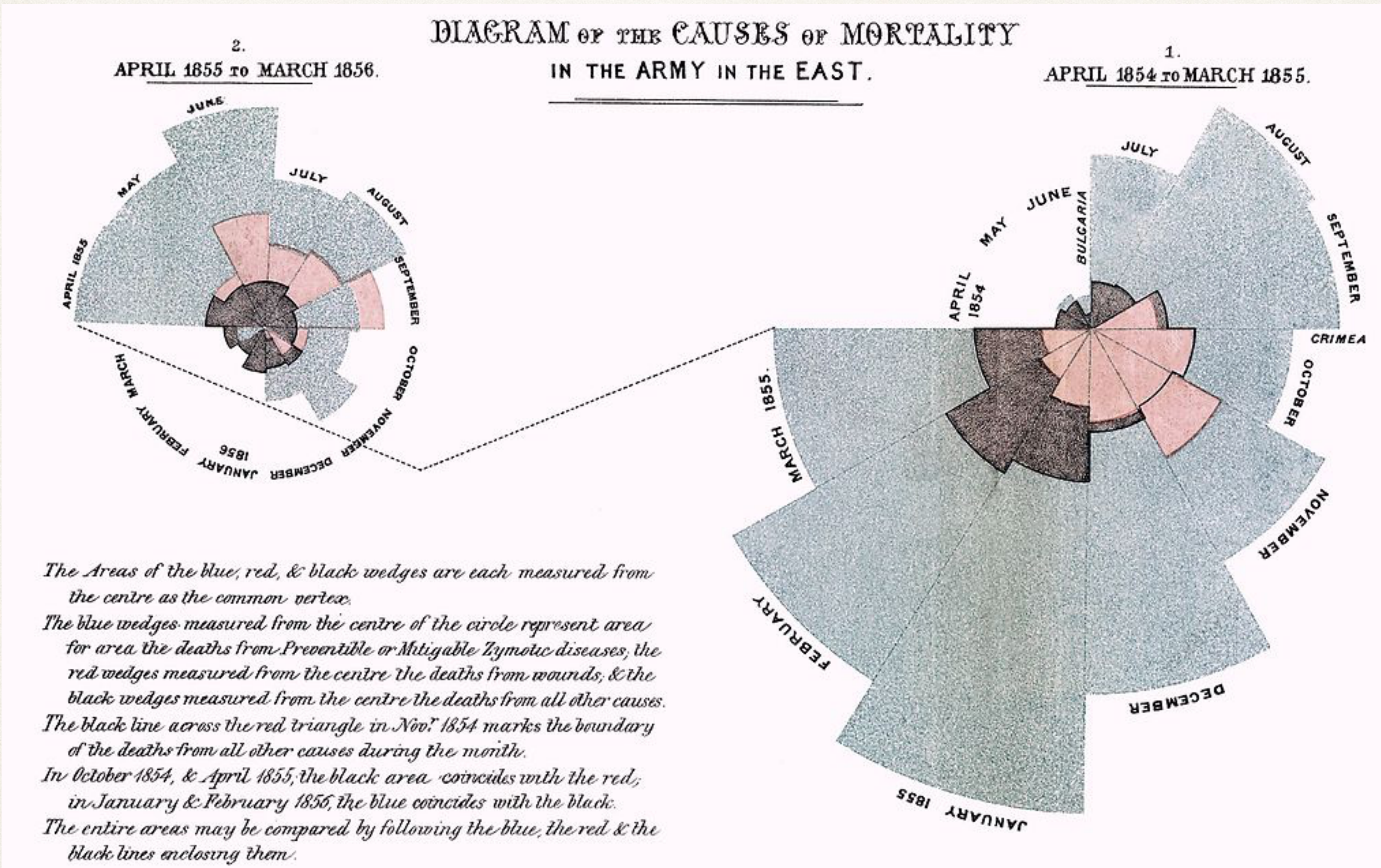
- In 1844 Charles Joseph Minard introduced the precursor of the modern mosaic plot (which we shall revisit) — note the variable width, divided (stacked) bars, where area is proportional to the quantity being shown



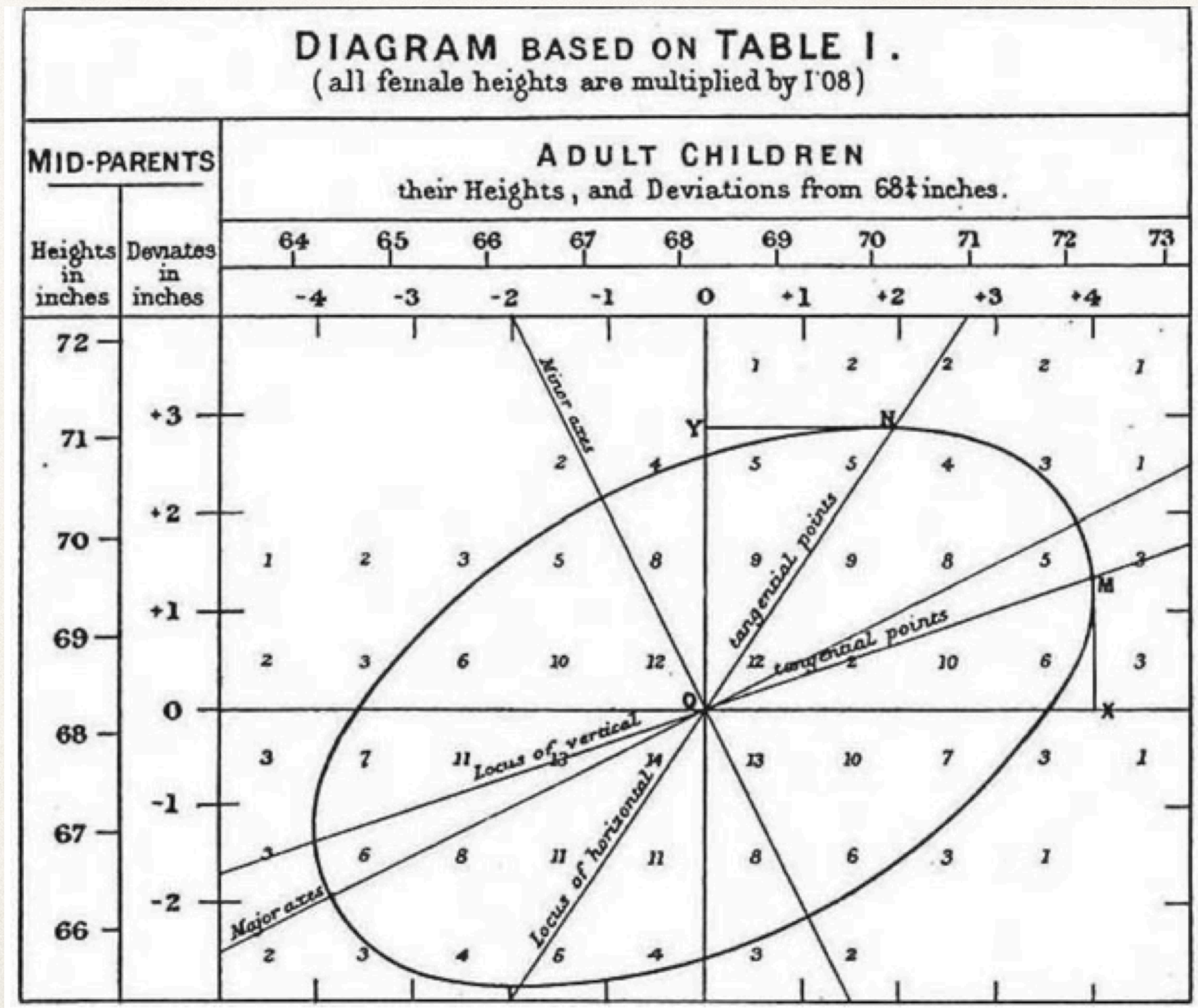
Minard’s visualisation of Napoleon’s Russian Campaign



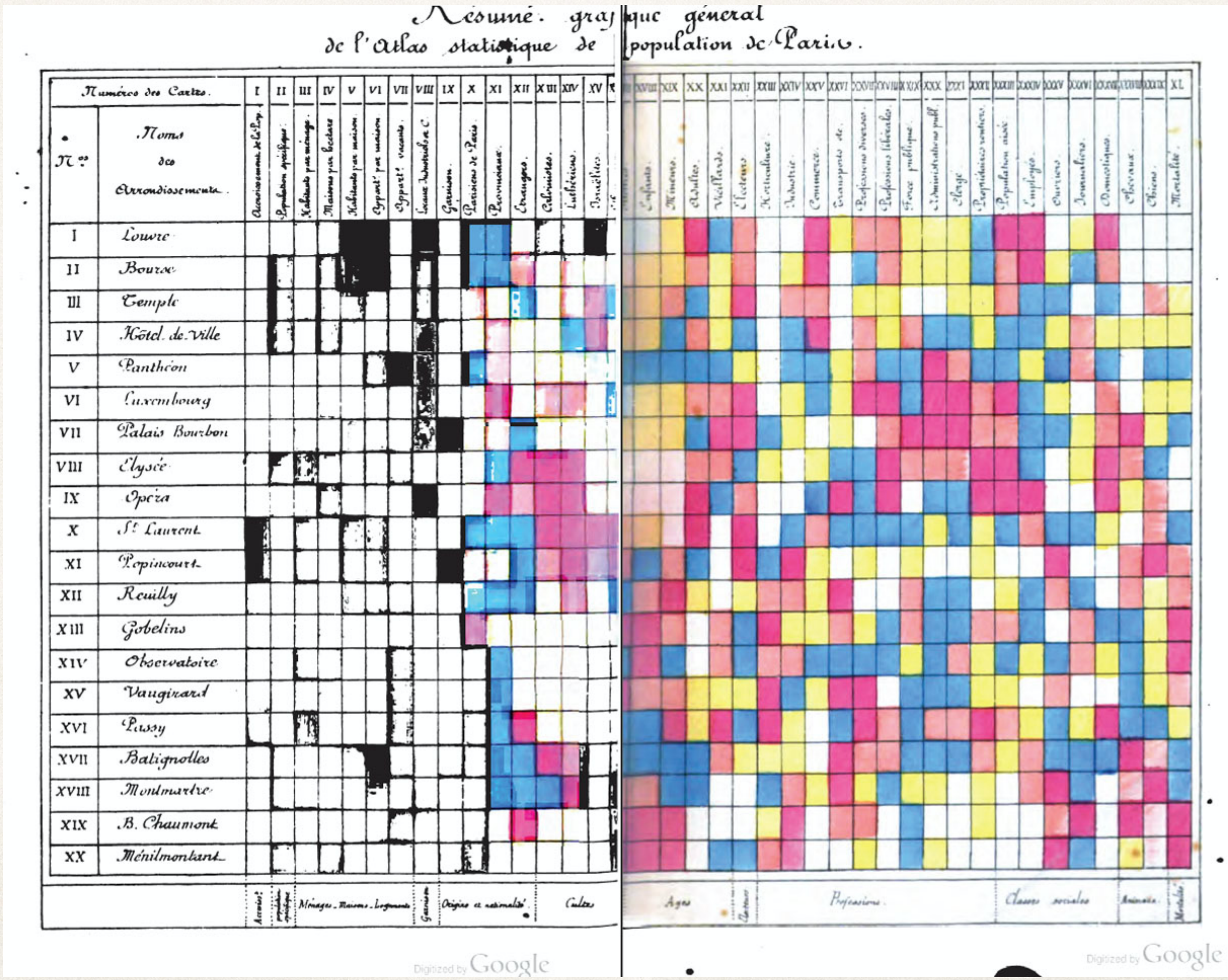
19th Century: Florence Nightingale's coxcomb charts



19th Century: Galton's bivariate normal plot, 1886



19th Century: *scalogram* by Toussaint Loua, 1873



These are now called ***heat maps***.

1900-1950: the modern Dark Ages

- Friendly describes this period as the modern Dark Ages for data visualisation
  - attributes it to the rise of quantification and the increasing use of formal statistical tests and models whose results could be reported in tables of coefficients and *p*-values
  - there was an increasing volume of statistical output, but in the pre-computer era, creating charts and other visualisations of these outputs was just too labour-intensive

1950-1975: the modern Renaissance of data visualisation

- In 1962, John Tukey published his landmark paper *The Future of Data Analysis*
  - in it, he introduced a wide range of new, simple but effective graphical displays under the rubric *Exploratory Data Analysis:* stem-leaf plots, boxplots, hanging rootograms, two-way tables displays
- In 1967, Jacques Bertin published the landmark monograph *Semiologie Graphique* which provided a theoretical framework for organising the visual and perceptual elements of graphics according to the features and relations present in data
  - we won't study Bertin in detail, but we will cover subsequent work by Cleveland and McGill, and Wilkinson, that builds on Bertin's work

1975 to present: the modern era

- Cheap, ubiquitous computing has revolutionised the everyday practice of data visualisation — we can do so much, and do it easily (but still visualisation is under-utilised). A few key developments:

  - visual data analysis using interactive, real-time linking, brushing, selection and focus (Becker and Cleveland, 1987)

  - visualising high-dimensional data: the grand tour (Asimov, 1985), scatterplot matrix (Tukey and Tukey, 1981), parallel coordinates plots (Inselberg, 1985; Wegman, 1990), spread plots (Young, 1994)

  - new and/or re-discovered methods for visualisation of discrete and categorical data: fourfold display (Fienberg, 1975), association plot (Cohen, 1980), mosaic plot (Hartigan and Kliener, 1981 & Friendly, 1994), sieve diagrams (Riedwyl and Schüpbach, 1983)

HDAT9800 Visualisation and Communication of Health Data

Chapter 1 - Part 2

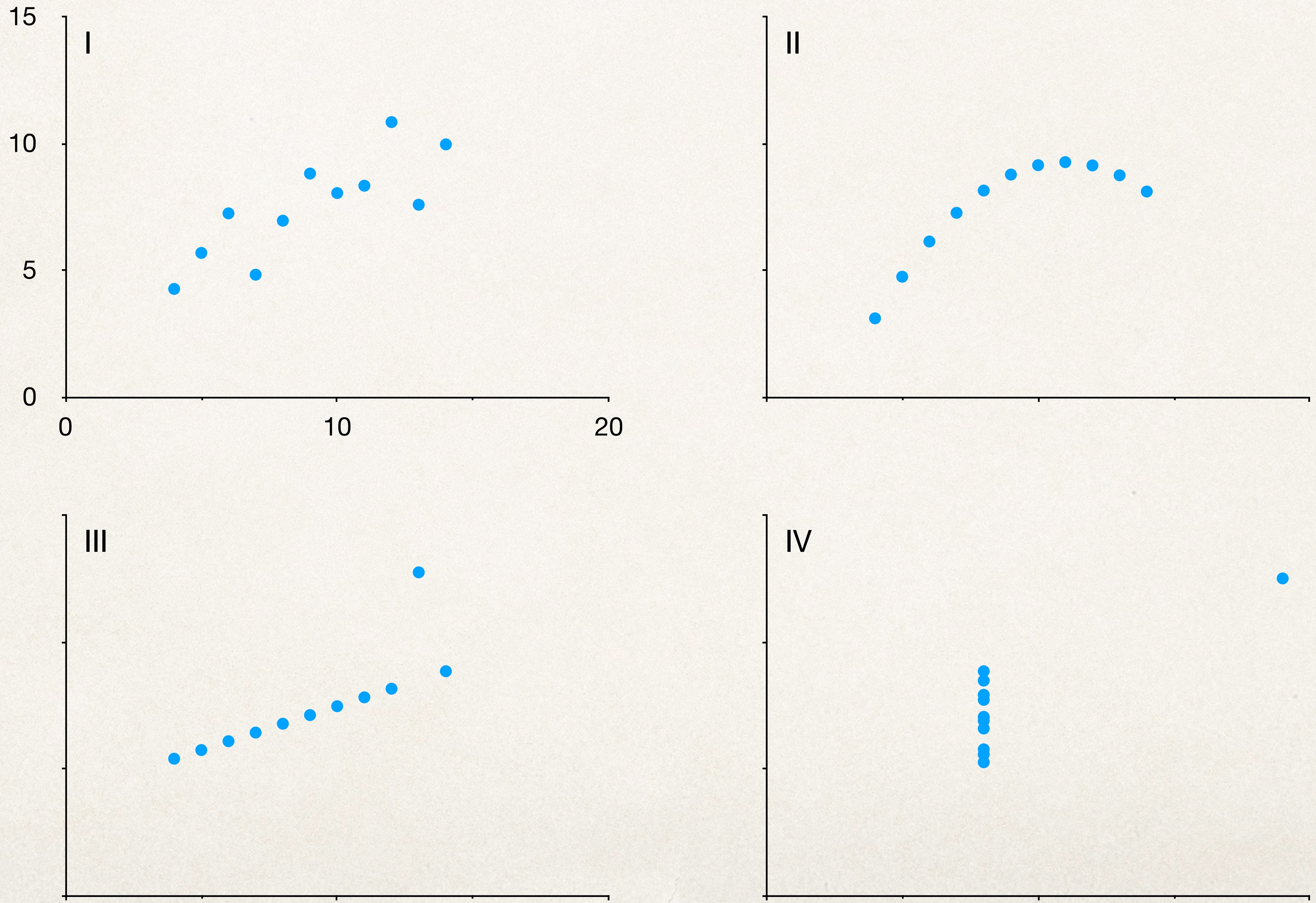
James Farrow & Tim Churches

Some general observations…

Tables or charts?

I		II		III		IV		Characteristics
X	Y	X	Y	X	Y	X	Y	
10.0	8.04	10.0	9.14	10.0	7.46	8.0	6.58	N = 11
8.0	6.95	8.0	8.14	8.0	6.77	8.0	5.76	mean(X) = 9.0
13.0	7.58	13.0	8.74	13.0	12.74	8.0	7.71	mean(Y) = 7.5
9.0	8.81	9.0	8.77	9.0	7.11	8.0	8.84	regression line: Y = 3 + X/2
11.0	8.33	11.0	9.26	11.0	7.81	8.0	8.47	std error of estimation of slope: 0.118
14.0	9.96	14.0	8.10	14.0	8.84	8.0	7.04	t = 4.24
6.0	7.24	6.0	6.13	6.0	6.08	8.0	5.25	sum of squares X - mean(X) = 110.0
4.0	4.26	4.0	3.10	4.0	5.39	19.0	12.50	residual sum of squares of Y = 13.75
12.0	10.84	12.0	9.13	12.0	8.15	8.0	5.56	correlation coefficient = 0.82
7.0	4.82	7.0	7.26	7.0	6.42	8.0	7.91	$r^2 = 0.67$
5.0	5.68	5.0	4.74	5.0	5.73	8.0	6.89	

Tables or charts?



The Datasaurus

```r
library(datasauRus)
if(requireNamespace("dplyr")){
  suppressPackageStartupMessages(library(dplyr))
  datasaurus_dozen %>%
    group_by(dataset) %>%
    summarize(
      mean_x    = mean(x),
      mean_y    = mean(y),
      std_dev_x = sd(x),
      std_dev_y = sd(y),
      corr_x_y  = cor(x, y)
    )
}
```

datasauRus package by Steph Locke, 2018, see https://cran.r-project.org/web/packages/datasauRus/vignettes/Datasaurus.html

The Datasaurus

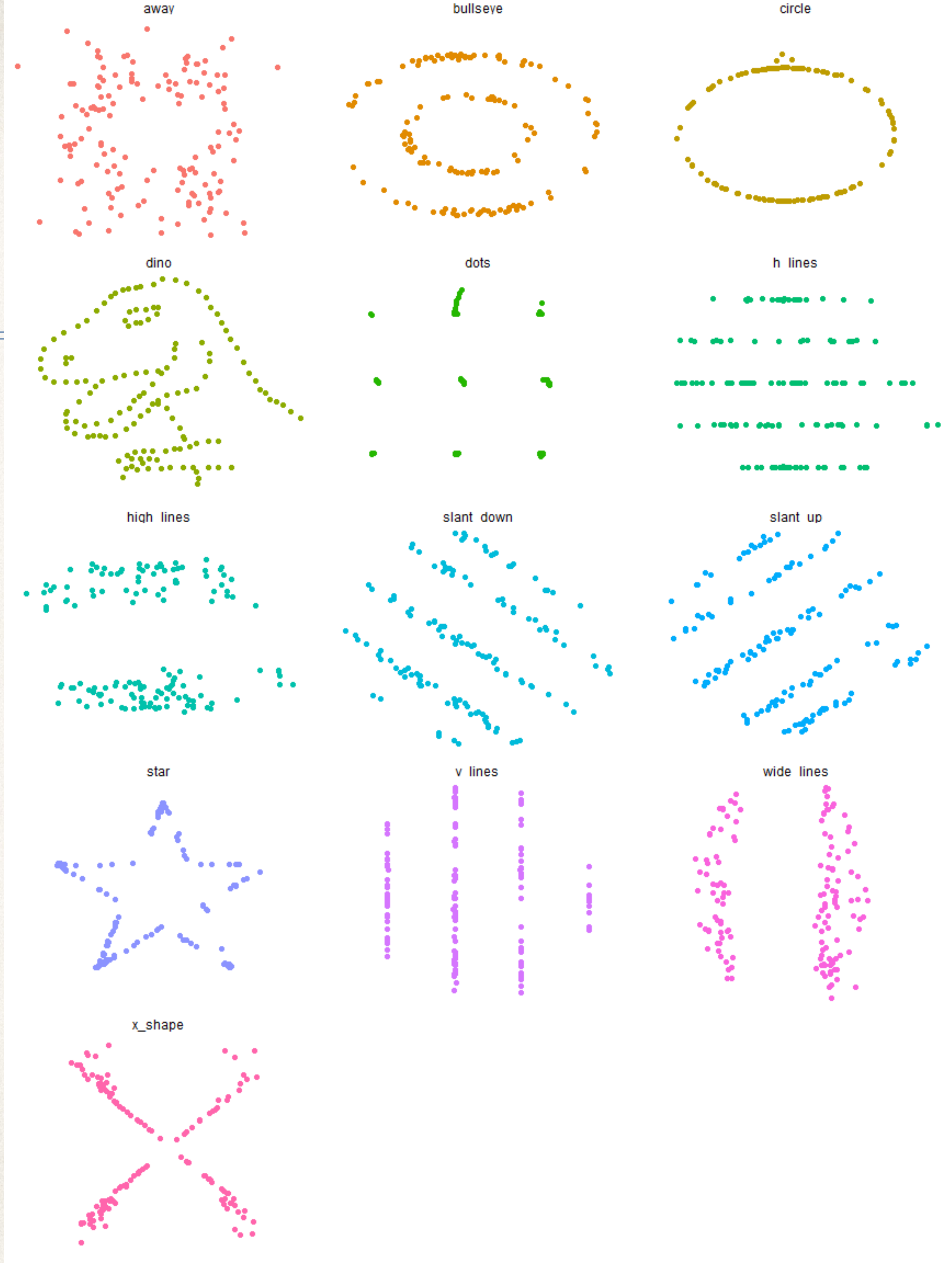
```
## # A tibble: 13 x 6
##    dataset    mean_x mean_y std_dev_x std_dev_y corr_x_y
##    <chr>       <dbl>  <dbl>     <dbl>     <dbl>    <dbl>
##  1 away         54.3   47.8      16.8      26.9  -0.0641
##  2 bullseye     54.3   47.8      16.8      26.9  -0.0686
##  3 circle       54.3   47.8      16.8      26.9  -0.0683
##  4 dino         54.3   47.8      16.8      26.9  -0.0645
##  5 dots         54.3   47.8      16.8      26.9  -0.0603
##  6 h_lines      54.3   47.8      16.8      26.9  -0.0617
##  7 high_lines   54.3   47.8      16.8      26.9  -0.0685
##  8 slant_down   54.3   47.8      16.8      26.9  -0.0690
##  9 slant_up     54.3   47.8      16.8      26.9  -0.0686
## 10 star         54.3   47.8      16.8      26.9  -0.0630
## 11 v_lines      54.3   47.8      16.8      26.9  -0.0694
## 12 wide_lines   54.3   47.8      16.8      26.9  -0.0666
## 13 x_shape      54.3   47.8      16.8      26.9  -0.0656
```

The Datasaurus

```r
if(requireNamespace("ggplot2")){
  library(ggplot2)
  ggplot(datasaurus_dozen, aes(x=x, y=y, colour=dataset))+
    geom_point()+
    theme_void()+
    theme(legend.position = "none")+
    facet_wrap(~dataset, ncol=3)
}
```

The Datasaurus

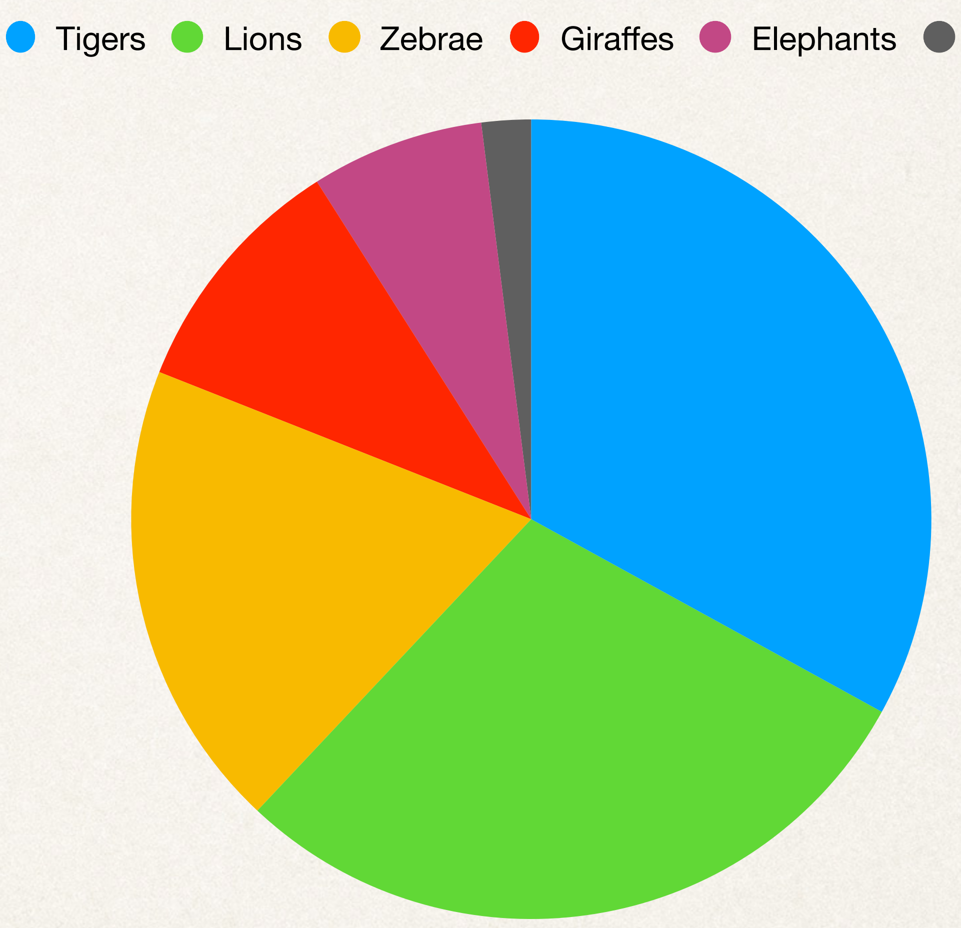
Always visualise your data!



Not all charts are equal

Are Tigers more popular than Lions?

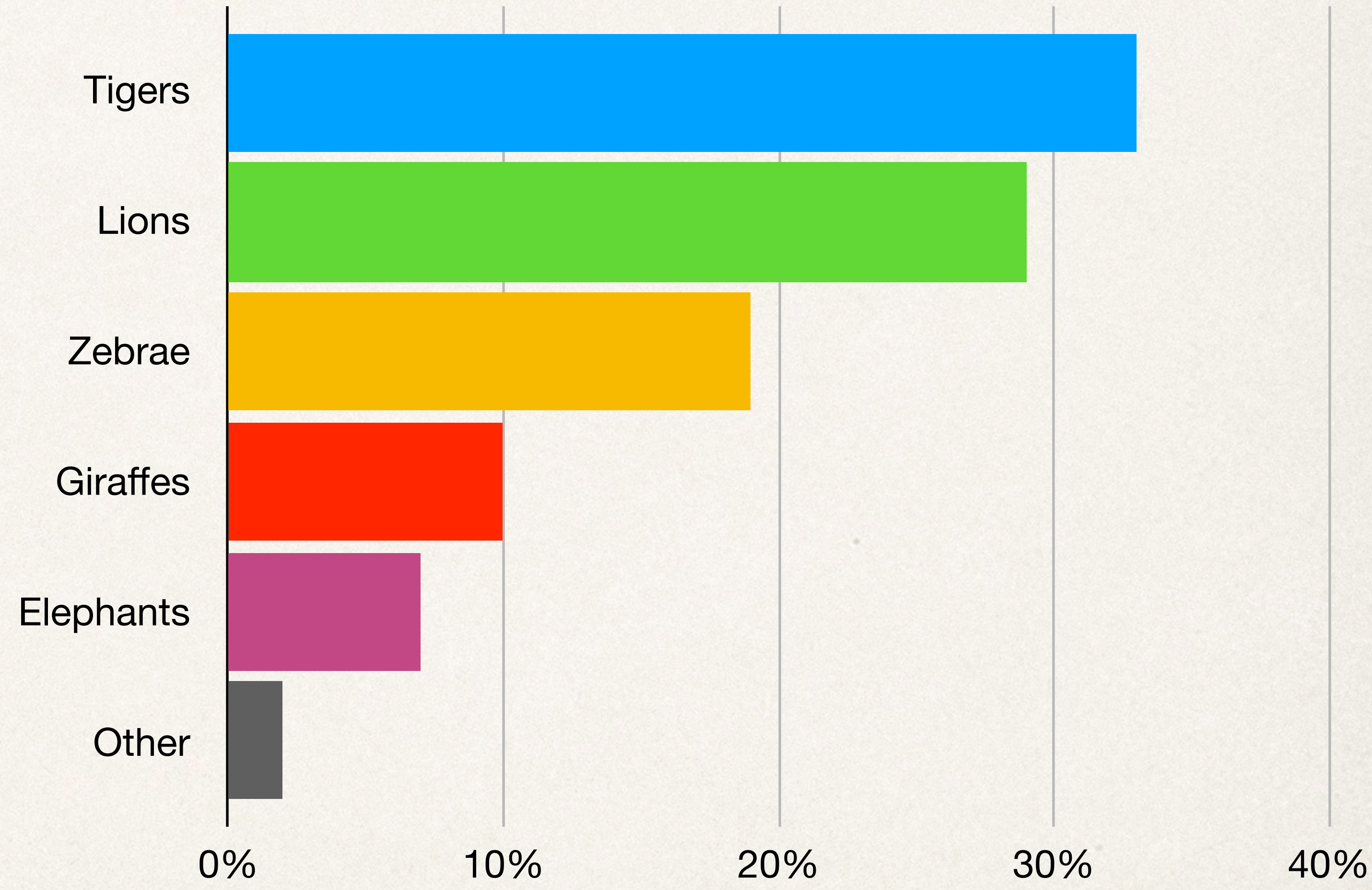
How much more popular are Zebrae than Giraffes?



Not all charts are equal

Are Tigers more popular than Lions?

How much more popular are Zebrae than Giraffes?



William Cleveland pointed out the problems with pie charts in the early 1980s. No-one listened…

Spatial information

How to lie with maps

Mark Monmonier. How to lie with maps. 3rd Ed. University of Chicago Press, Chicago. 2018. ISBN 9780226435923

https://press.uchicago.edu/ucp/books/book/chicago/H/bo27400568.html

There is no magic bullet

Different presentations are appropriate in different contexts

Sometimes we need a wow! factor

Sometimes we need accurate numbers

Sometimes we need to present just one thing or correlate many things

Sometimes we have 3 minutes to talk, sometimes 30

Sometimes we are talking and sometimes we are writing

Sometimes we are writing a paragraph or a paper or a book

Some General Principles

Clarity

- More is not necessarily better

Reproducibility

- You need to be able to repeat your work
- Other people need to be able to repeat your work

Collaboration

- Work effectively with others (including your future self)

The point is communication

The aim is to *communicate*

Not necessarily *as much as possible* or *as fast as possible*

Overwhelming people with information can reduce communication

Showing working can help develop a mental model

Showing working can confuse and overwhelm

Know your audience

Stay on point

Communicate the necessary things clearly

Pointers to extra material can be provided

Deeper working/analysis can be an addendum or provided as links

Reproducibility crisis

A problem of our age

Many studies are difficult or impossible to replicate or reproduce

According to a 2016 poll of 1,500 scientists reported in the journal *Nature*, 70% of them had failed to reproduce at least one other scientist's experiment (50% had failed to reproduce one of their own experiments)

Field	Other	Self
Chemistry	87%	64%
Biology	77%	60%
Physics and Engineering	69%	51%
Earth Sciences	64%	41%

Reproducibility crisis

Reproducibility is used in two senses:

- the first sense refers to the large number of under-powered studies which happened (by chance) to have produced statistically significant results
- big problem in biomedical ("wet lab") research and in psychological experiments
- growing problem in clinical research

Solution: don't perform under-powered studies, and/or report FDRs (false discovery rates)

Reproducibility crisis

Second sense of *reproducibility*

- Inability to repeat, or even explain, all the steps taken to go from the raw research data to the final analysis results.
- This is the sense we address in this course (both senses are important!)

Ensure our data manipulation and analysis is *repeatable* and *transparent*

- repeatable: automated *versus* manual
- transparent: make working available

Reproducibility (second sense)

Time/space constraints may limit options

- only appropriate for some media types
- may help a mental model of process and resulting artefacts

Allows verification of method

Allows replication on other data

Collaboration

Collaboration is important

You will rarely work alone

Most workflows and tools ill-equipped to manage it well

We will be looking at *git*

- Suited to online and offline development
- Suited to individual and group work
- Better suited to textual artefacts but will cope with anything

Core best practices for reproducibility and collaboration

- Always keep a record of development
- Never alter original data
- Eliminate manual steps
- Develop a programmatic pipeline of transformation/analysis
- Open, transparent and reproducible processes

Toolbox approach

A 'starter kit' of useful approaches and ideas to build on

- *git*
- *ggplot2* (and *dplyr*)
- Rmarkdown and *knitr*
- *leaflet*
- *shiny*

Introduction to R

Introductory tutorial if you're rusty or want to ease in to R

A *learnr* module

- runs in your browser
- interactive: can enter R code and execute it

Will need the *learnr* R package installed

Version 1.3 or later of RStudio is **strongly** recommended

- v1.3 makes launching *learnr* tutorials much easier

Introduction to R

Six sections

- overview of R and basic types, arithmetic, logic and comparison
- vectors and subsetting
- matrices and lists
- factors, data frames and reading from CSV files
- writing functions and variable scope
- control flow: if/else, while, for, repeat

R packages: CRAN

The *Comprehensive R Archive Network (CRAN)* is a repository of third party R packages

It is a network of servers around the world that store identical, up-to-date, versions of code and documentation for R

We will be looking at and using several third party R packages available through CRAN

- although some of the packages we examine in the interactive sessions may need to be installed directly from GitHub (instructions will be provided)

Using *dplyr*

Basic R is flexible but can be rather unreadable when subsetting data

- there are many ways to do the same thing, which can be confusing, and quite a few inconsistencies and “gotchas” that trip up many user

The *dplyr* package makes this process more readable

Allows the creation of a pipeline of manipulation

Original data frame remains unchanged

A new data frame is produced

dplyr tutorial

Installed as a package like *Introduction to R*

Needs some other packages to be installed

- *dplyr*
- *nycflights13*
- *ggplot2*

Install these using the package manager in RStudio

Literate programming

A concept developed by one of the giants of computer science, Donald Knuth

Explanation of logic in natural English interspersed with code fragments

In other words ‘a report with embedded code’

The *knitr* R package implements this approach

- Jupyter for python is similar

RStudio is a nice IDE supporting this paradigm

Literate programming

Writing documents and code *as you go*

Report and code development tied together

- the write-up *is* the analysis, the analysis *is* the write-up

Ensures accuracy and reduces barriers to completion

Well suited to individual and collaborative report writing

- text format ideal for use with *git*
- makes incremental refinement easy

Literate programming

We'll explore this in the context of individual report writing

We'll revisit it in the context of group collaboration

We'll see how *git* helps manage versions in both contexts

Using Rmarkdown and *knitr*

Simple markup to format text

Fits most simple formatting needs

Used by *knitr* to create reports interleaving text and code

Enables creation of a document encapsulating reproducible processes

Transparency

Because the report *is* the code it can be distributed

‘Here is *exactly* what I did to produce the written report’

Recreate the report/analysis

Extend or modify the analysis

Recreate the analysis on a different data set

Using *git*

Used to manage personal workflow and development

Scales gracefully to multiple member teams (even huge teams)

Allows disconnected operation

- you can work away from the internet
- work on a plane or while travelling
- upload/download resync once you're back

Extremely flexible workflow and sharing mechanisms

Versioning using *git*

Will store and manage versions of anything

Most powerful with text-based documents

- Rmarkdown, latex, code, …
- fine-grained changes in documents

Other artefacts will be stored whole and managed

Enables comparison and rollback to previous points in time

Something has broken? When did it break? What changed?

Using *ggplot2*

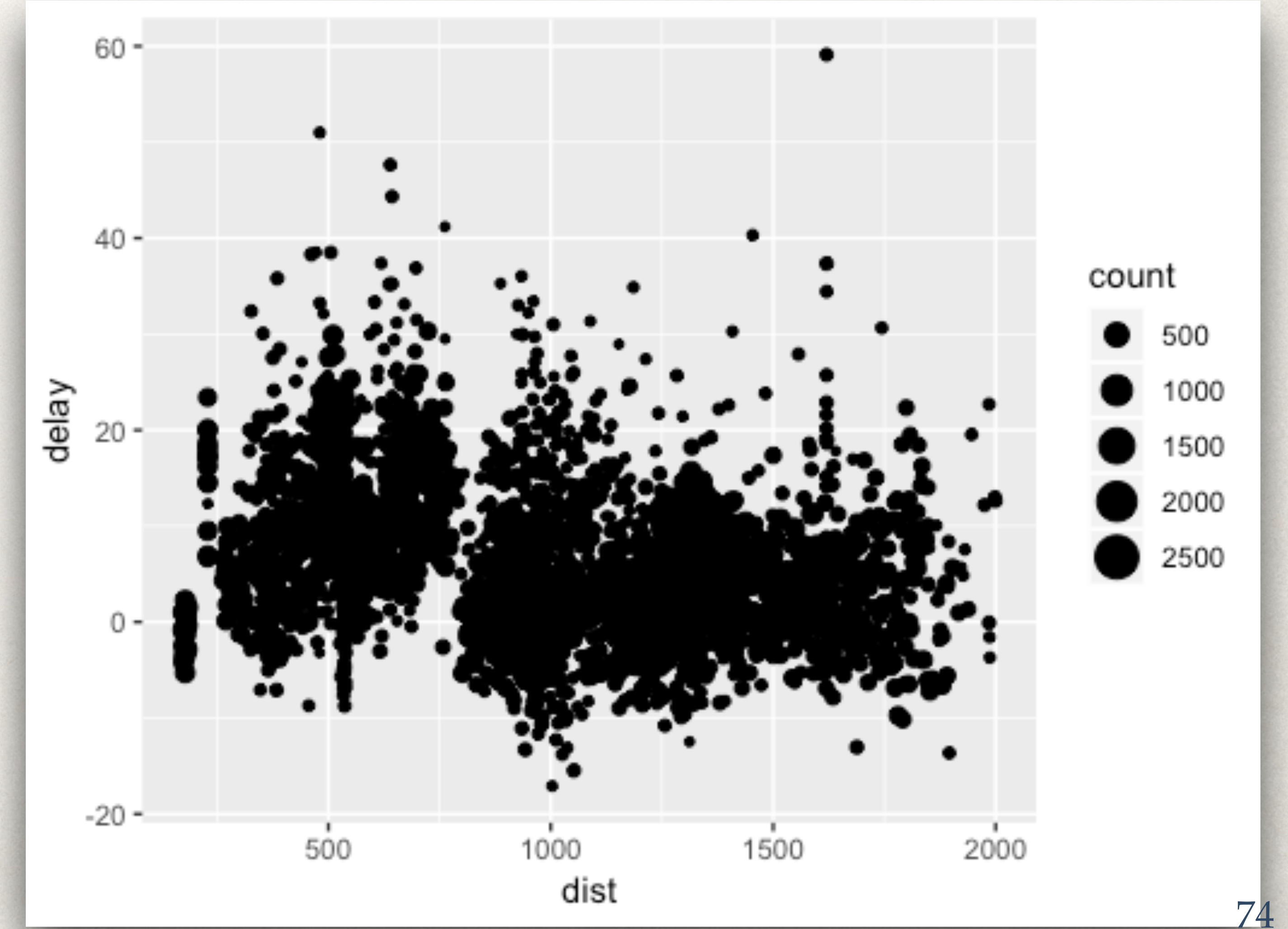
A 'gold standard' of charting for R

Built on Wilkinson's *grammar of graphics* (the 'gg' bit)

- specify data
- specify aesthetic mapping
- specify geometries (the way the data is shown)
- specify tweaks

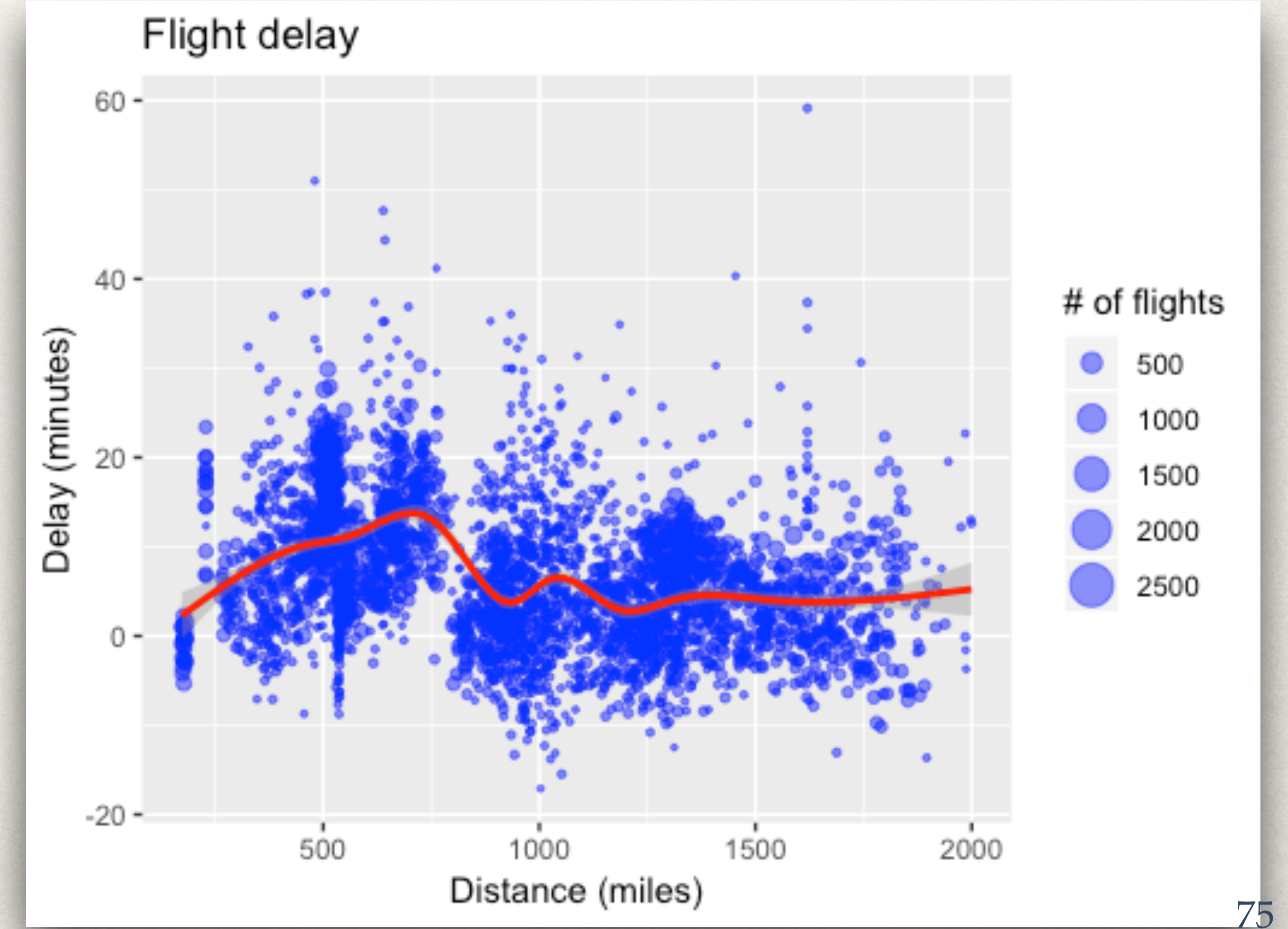
Using *ggplot2*

```
ggplot(delay, aes(dist, delay)) +
    geom_point(aes(size = count), alpha = 1/2)
```



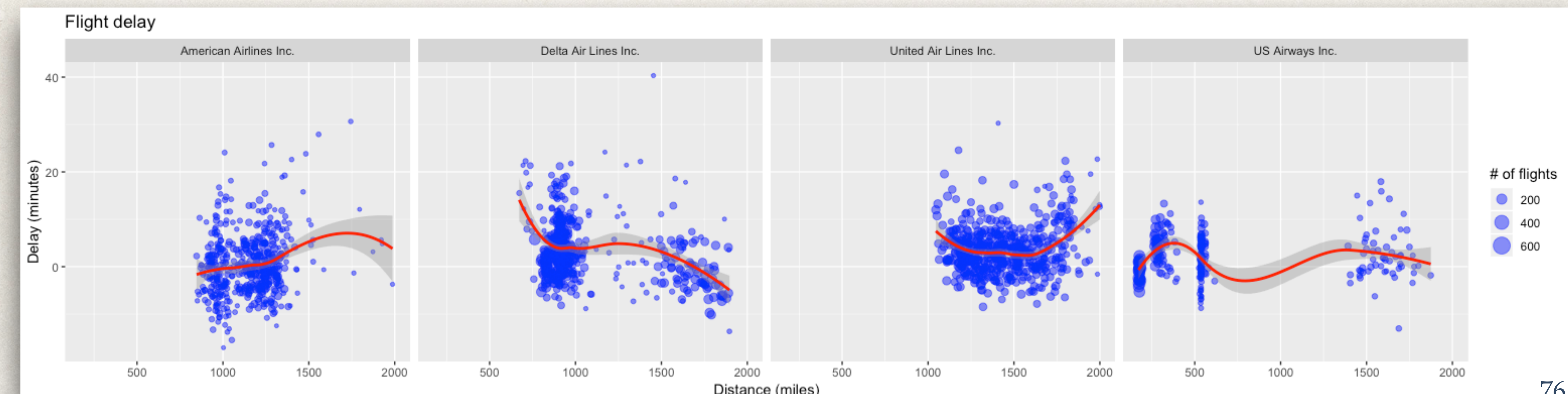
Using *ggplot2*

```
ggplot(delay, aes(dist, delay)) +
    geom_point(aes(size = count), alpha = 1/2, color="blue") +
    geom_smooth(color="red") + scale_size_area() +
    labs(x="Distance (miles)",
         y="Delay (minutes)",
         size="# of flights") +
    ggtitle("Flight delay")
```



Using *ggplot2*

```
ggplot(delay, aes(dist, delay)) +
    geom_point(aes(size = count), alpha = 1/2, color="blue") +
    geom_smooth(color="red") + scale_size_area() +
    labs(x="Distance (miles)",
         y="Delay (minutes)",
         size="# of flights") +
    ggtitle("Flight delay") + facet_grid(. ~ name)
```



Collaboration using *git*

The text based format of programming code lends itself to collaboration

git allows work on *different parts* of the same document at the same time

git allows work on the *same parts* of the same document at the same time

Conflicting versions flagged and managed for reconciliation

Nothing is every overwritten or lost

No ‘global lock’ holding up work

Collaboration using *git*

We'll be using *git*-based collaboration for the group assignment

Group members will have a common repository on *GitHub* to share changes

Work independently or together

Publish and merge changes from time to time as desired

Version control is about *managing changes*

Once we know what the changes are we can share our changes

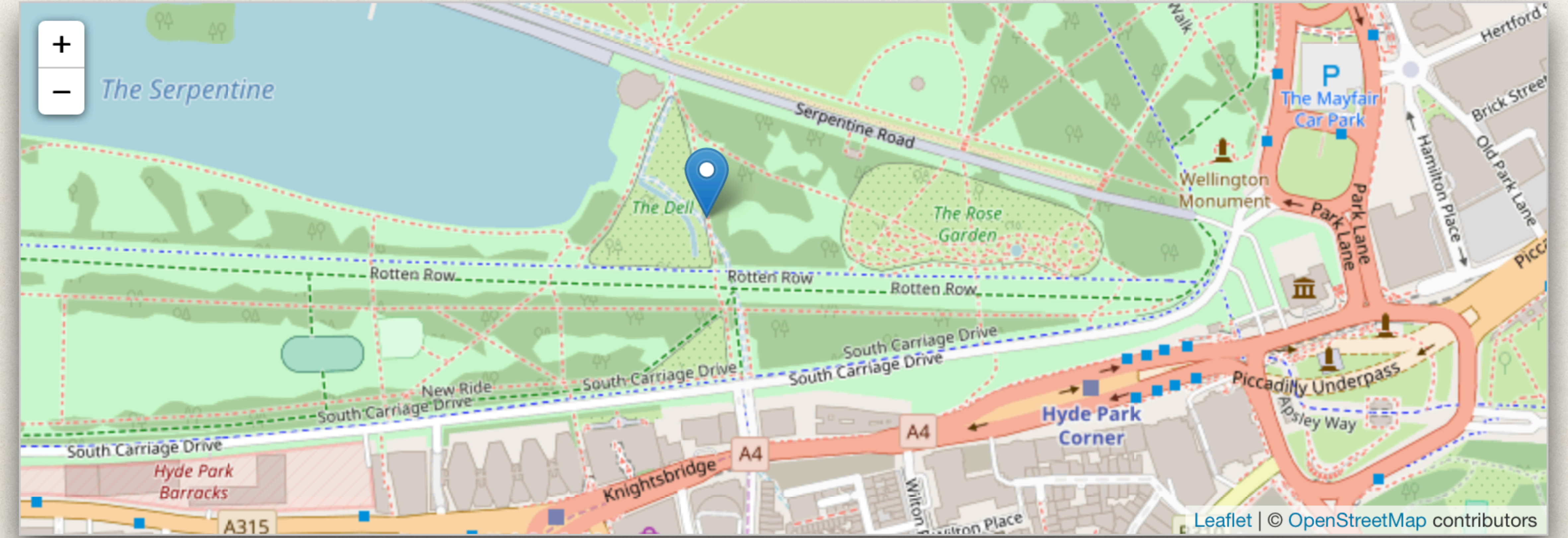
We don't have to share everything, just changes

Using *leaflet*

An open-source JavaScript library for interactive maps

- *https://leafletjs.com*

4 lines of R code gives us something like this



Using *leaflet*

We'll be looking at how we can use *leaflet* from R

Overlaying our own features of interest

Overlaying boundaries and regions onto existing maps

Interactively highlighting and colouring map features

Using *shiny*

Interactive web-based applications written in R

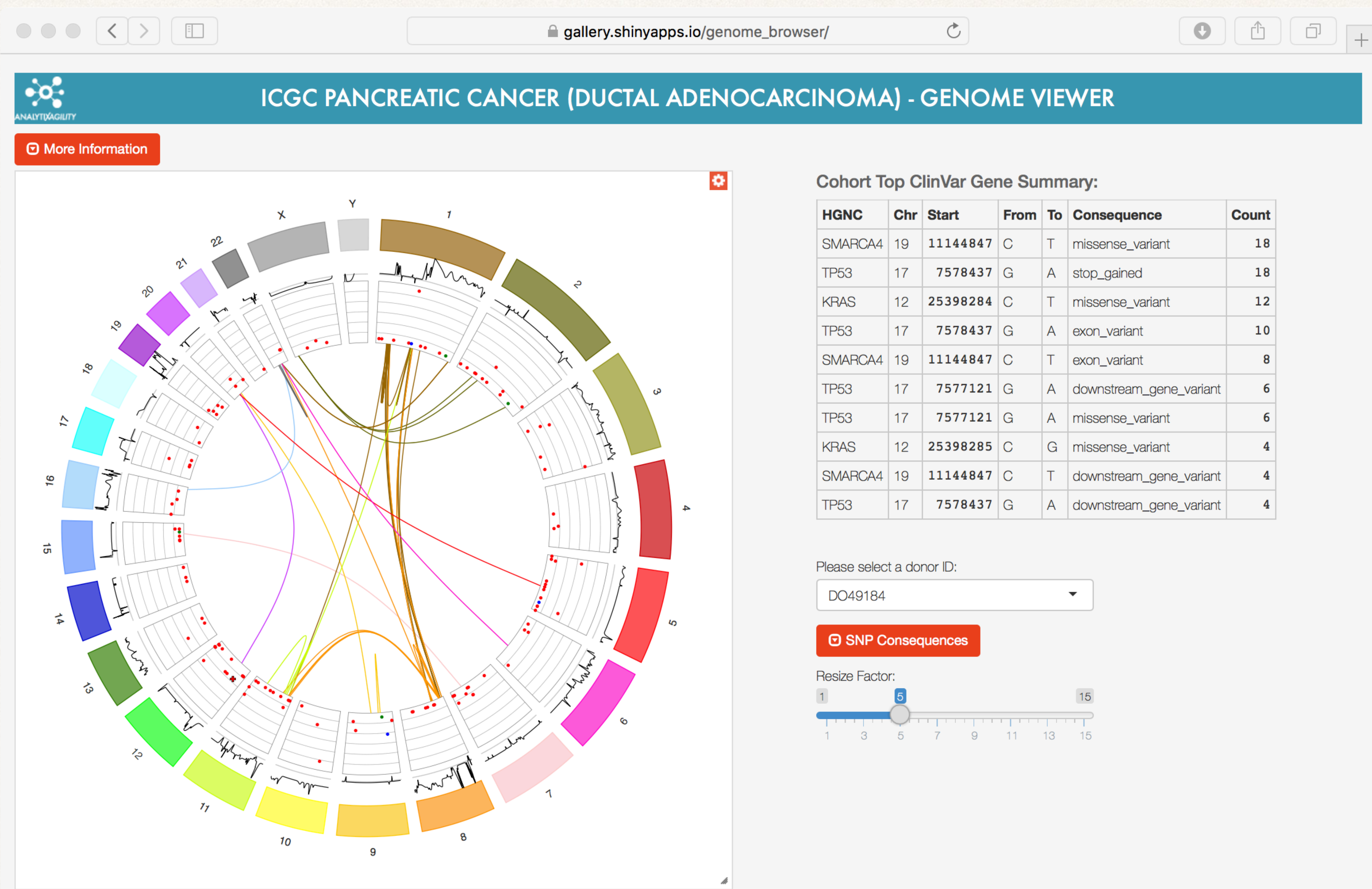
The *learnr* tutorials we'll be using for this course are *shiny* apps

Hides the details of HTML, CSS and JavaScript for you

Shiny showcase at

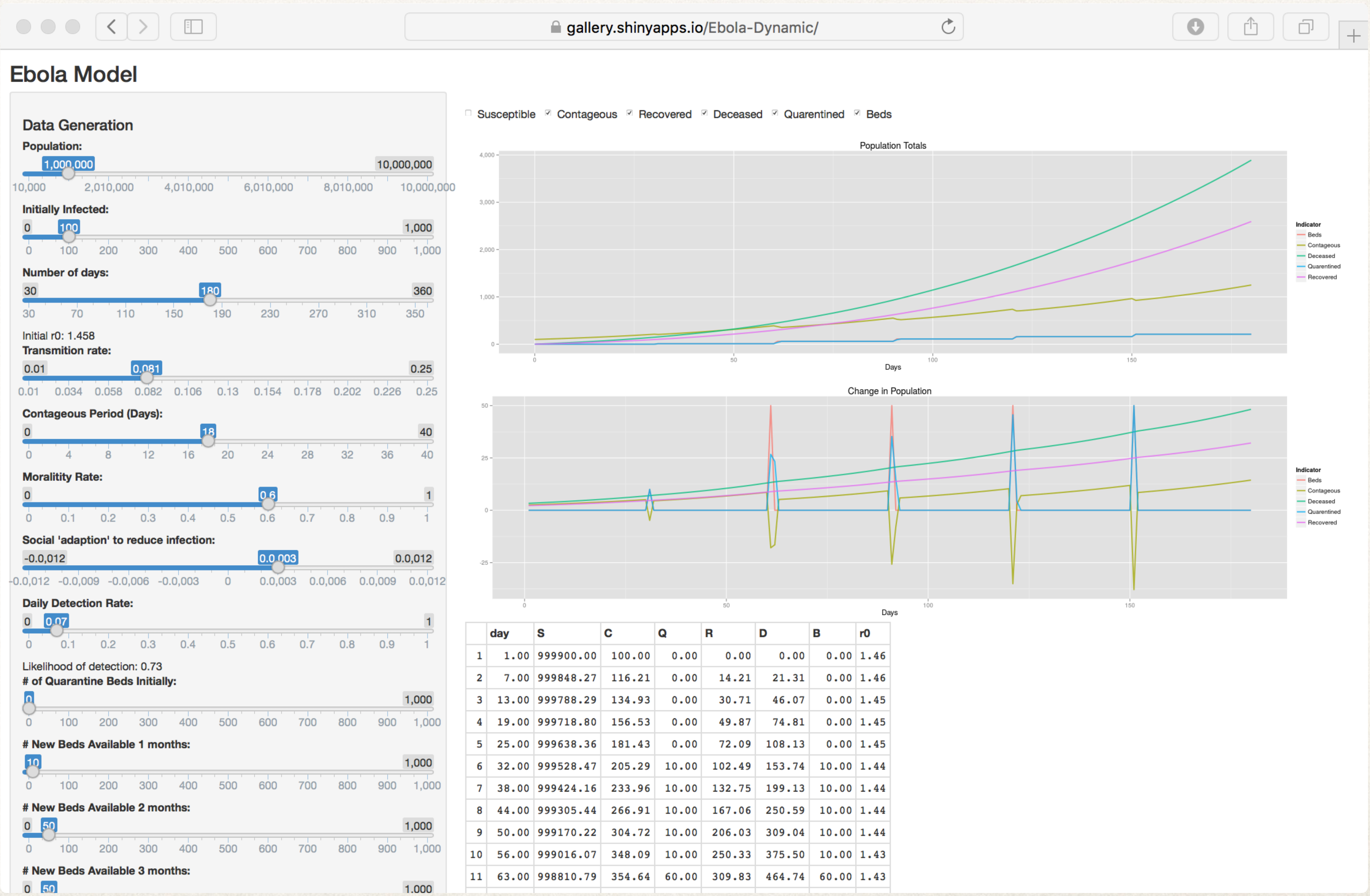
- *https://www.rstudio.com/products/shiny/shiny-user-showcase/*

Genome browser



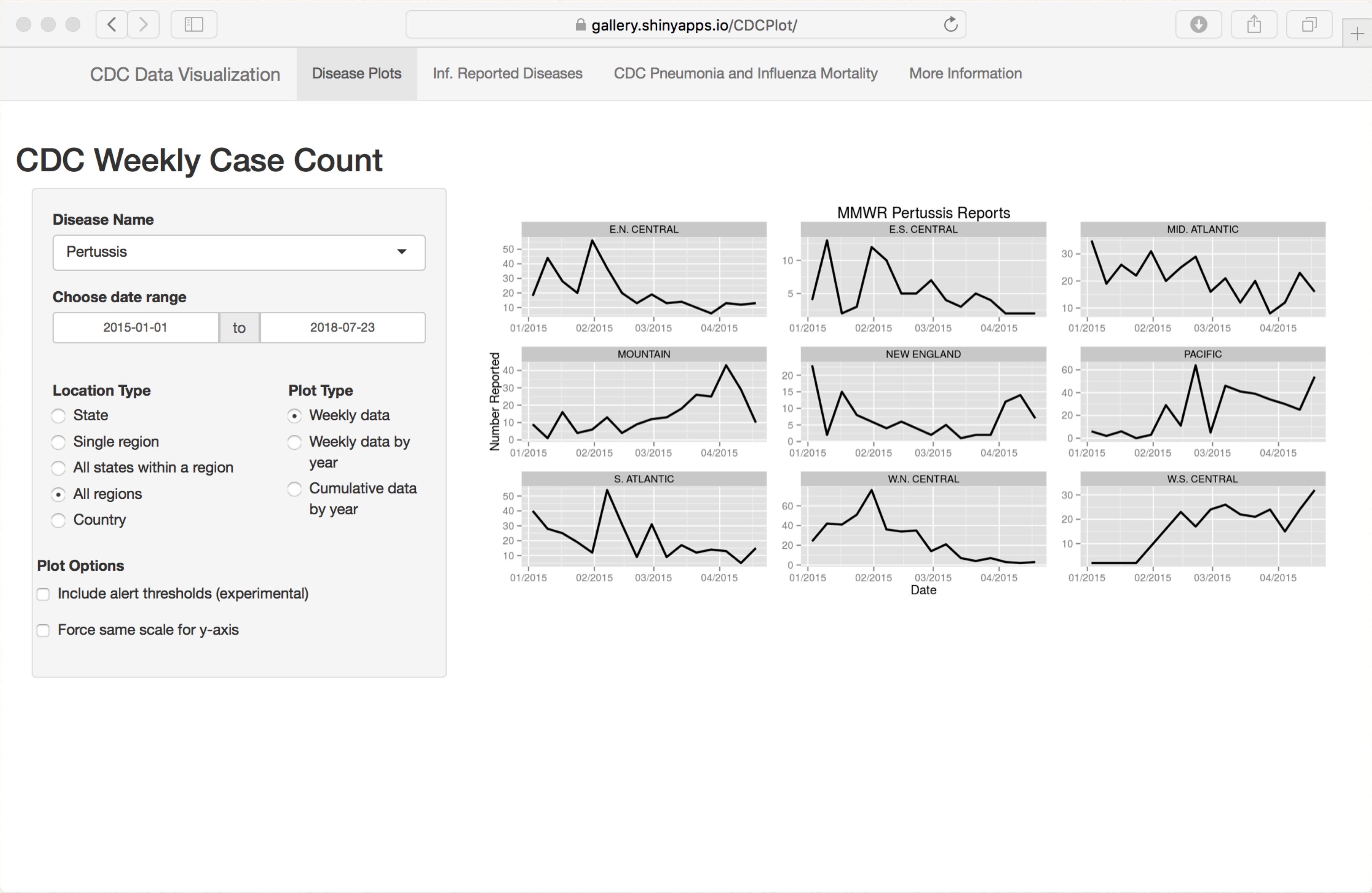
HGNC	Chr	Start	From	To	Consequence	Count
SMARCA4	19	11144847	C	T	missense_variant	18
TP53	17	7578437	G	A	stop_gained	18
KRAS	12	25398284	C	T	missense_variant	12
TP53	17	7578437	G	A	exon_variant	10
SMARCA4	19	11144847	C	T	exon_variant	8
TP53	17	7577121	G	A	downstream_gene_variant	6
TP53	17	7577121	G	A	missense_variant	6
KRAS	12	25398285	C	G	missense_variant	4
SMARCA4	19	11144847	C	T	downstream_gene_variant	4
TP53	17	7578437	G	A	downstream_gene_variant	4

Ebola Outbreak Simulation

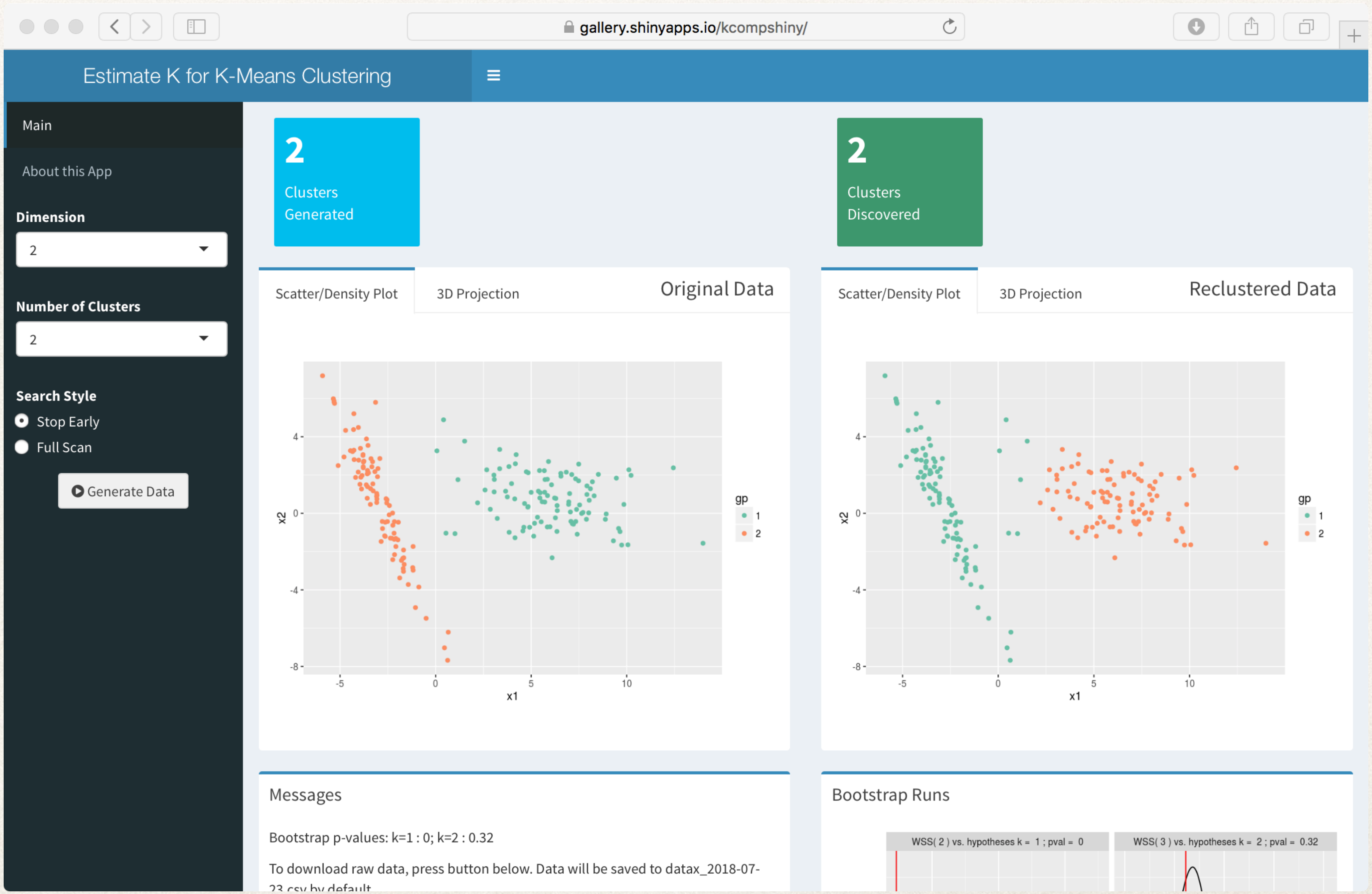


	day	S	C	Q	R	D	B	r0
1	1.00	999900.00	100.00	0.00	0.00	0.00	0.00	1.46
2	7.00	999848.27	116.21	0.00	14.21	21.31	0.00	1.46
3	13.00	999788.29	134.93	0.00	30.71	46.07	0.00	1.45
4	19.00	999718.80	156.53	0.00	49.87	74.81	0.00	1.45
5	25.00	999638.36	181.43	0.00	72.09	108.13	0.00	1.45
6	32.00	999528.47	205.29	10.00	102.49	153.74	10.00	1.44
7	38.00	999424.16	233.96	10.00	132.75	199.13	10.00	1.44
8	44.00	999305.44	266.91	10.00	167.06	250.59	10.00	1.44
9	50.00	999170.22	304.72	10.00	206.03	309.04	10.00	1.44
10	56.00	999016.07	348.09	10.00	250.33	375.50	10.00	1.43
11	63.00	998810.79	354.64	60.00	309.83	464.74	60.00	1.43

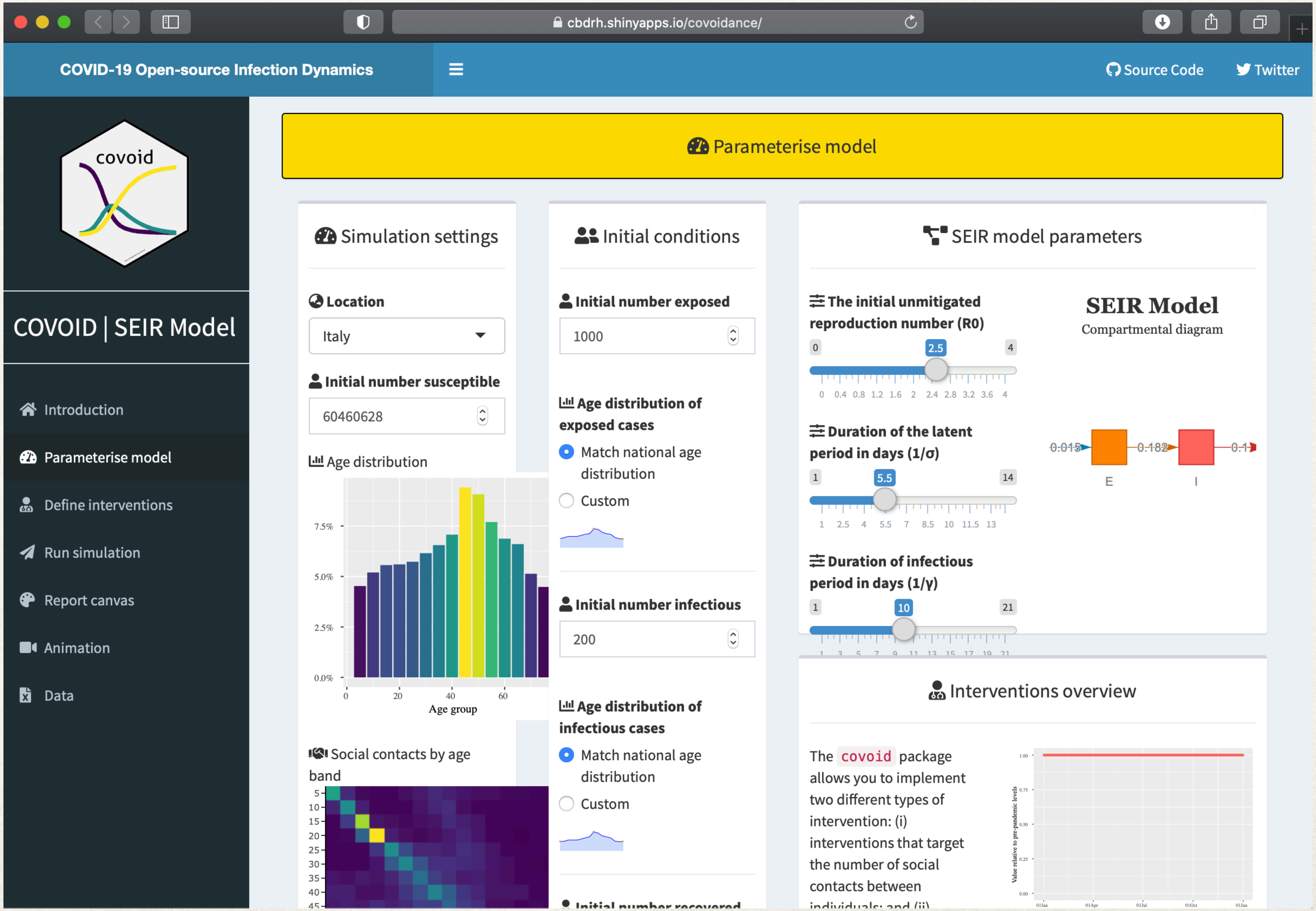
CDC Disease Monitor



Bootstrap Clustering



Modelling COVID-19 transmission



Modelling COVID-19 transmission

cbdrh.github.io/covoidance/

UNSW SYDNEY COVOID Documents ▾ Shiny app Acknowledgements

COVOID

The **COV**ID-19 **O**pen-source **I**nfection **D**ynamics project

eRum2020::CovidR Winner

COVOID is a R package for modelling COVID-19 and other infectious diseases using deterministic compartmental models (DCMs). It contains a built-in Shiny app enabling easy use and demonstration of key concepts to those without R programming backgrounds, along with an expanding API for simulating and estimating homogeneous and age-structured SIR, SEIR and extended models. In particular COVOID allows the simultaneous simulation of age specific (e.g. school closures) and general interventions over varying time intervals.

covoid

Please see the documentation available from the menu above for more information.

Acknowledgements and support

Work on COVOID is facilitated by the generous assistance of **Ian Sharp**, philanthropic supporter of UNSW research.

https://cbdrh.github.io/covoidance/

shiny is extensible

Access to underlying HTML, CSS and JavaScript

- everything can be customised

Apps can be embedded in other Rmarkdown documents or in a dashboard of apps

Lots of written and video tutorials online

shiny and the group assignment

The group assignment is to write a simple interactive analysis with geospatial and (simple) statistical aspects as a *shiny* app

Development will be done in RStudio using *git*, *shiny*, *leaflet*, *ggplot2* and GitHub Classroom

By the end of the course

Practical life skills for health data scientists

- Understand better what comprises 'good' and 'bad' visualisations
- Be comfortable programming in R for data manipulation and visualisation
- Be confident using *git* to manage versioning and development
- Be aware of the features of *ggplot2* and be able to use them
- Be able to collaborate using Rmarkdown and *knitr*

Mission: POSSIBLE

Do the two tutorials for this week

- *Introduction to R*
- *dplyr*

Look over the first weekly assessment

- simple questions
- designed to get you comfortable with the assignment submission process