

Data Visualisation and Visual Demography   
using R and ggplot 2

22nd – 23rd October 2014  
Grosvenor Hilton, Glasgow

Written and presented by:  
Dr Jonathan Minton, University of Glasgow

For additional material and updates please visit:

<https://github.com/JonMinton/AQMEN_Data_Vis_Workshop>

© Copyright   
  
Dr Jonathan Minton, University of Glasgow.  
  
This document has been produced for AQMeN by Dr Jonathan Minton and edited by Heather Thomson.  
  
Any material in this file must not be reproduced, published or used for teaching without permission from the author.  
For information contact [info@aqmen.ac.uk](mailto:info@aqmen.ac.uk)

Contents

[1. Take-Home Messages 1](#_Toc400548799)

[2. The Grammar of Graphics 3](#_Toc400548800)

[3. The Layers of Graphics 5](#_Toc400548801)

[4. The Three Hats of Data Visualisation 7](#_Toc400548802)

[5. Table Management 9](#_Toc400548803)

[6. References 11](#_Toc400548804)

[6.1 Books 11](#_Toc400548805)

[*6.1.2 Graphics in R using ggplot2 and lattice* 13](#_Toc400548806)

[6.2 Websites 14](#_Toc400548807)

[6.3 Software 14](#_Toc400548808)

# Take-Home Messages

The main take-home messages of the workshop are:

1. Data visualisation must involve *the consistent mapping of variables within a dataset to aesthetic elements of an image*.
2. Data visualisations usually include a number of distinct layers, defined by the role they play in conveying information to the viewer. At its simplest, there are three layers: the **support layer**, the **data layer**, and the **annotation layer**. What makes a data visualisation a data visualisation, rather than some other kind of information visualisation, is the presence of the data layer.
3. There is a **grammar of graphics**, a way of describing the different elements that go into producing a data visualisation. Learning some of these terms, even if we don’t apply them consistently, can help us think and communicate more clearly about the elements, processes and aims of producing a visualisation.
4. You need three distinct mind-sets for data visualisation, three ‘hats’ that you need to swap between in order to complete different tasks involved in the process. These are the **Scientist**, the **Artist**, and the **Engineer**. On a day-to-day basis, you will spend most of your time wearing the Engineer’s hat. (Also, Scientists and Artists tend to squabble.)
5. *What counts as a good data visualisation is heavily context dependent*. A good data visualisation is one that is a good match for the audience. Audiences can be divided crudely into ‘internal’ and ‘external’ (we and they); and into ‘specialist’ and ‘generalist’. Often but not always, internal audiences are specialists, and external audiences are generalists. Data visualisations meant for internal audiences are visualisations we produce for ourselves. In these cases, visualisations are used as part of exploratory data analysis, to learn something new. By contrast, data visualisations meant for external audiences aim to communicate something we already know to other people.

# 2. The Grammar of Graphics

Grammar is a set of tools for describing the hidden structure of sequences of words. Just as there is a grammar of language, so there is a grammar of graphics. The grammar of language distinguishes words and sequences as adjectives, nouns, pronouns, propositions, adverbs, and so on, based on the roles and functions words and sequences have in the context of sentences and paragraphs. Similarly, the grammar of graphics distinguishes different pieces of a data visualisation according to their functional components.

A book, **The Grammar of Graphics**, was written by the statistician Leland Wilkinson in 2005. The book represents a highly formalized approach to understanding what goes into producing a data visualisation, providing terms and ideas for helping researchers distinguish between different component parts of a visualisation. The components of the grammar of graphics that Wilkinson identifies include:

* **Variables**: these are pieces of data, associated with observations, which contain values.
* **Scales**: these are choices of transformations that can be applied to the values associated with variables. For example, an identity scale produces as output whatever was given as its input, and a logarithmic transformation produces as output the logarithm of the input.
* **Geometries**: these are specific shapes and forms which you, as producer of the data visualiser, must choose between. Your choice will depend on your understanding of the types of variables being represented, and the types of relationship between them. For example, a line is a different geometry to a point, and a line which connects two points is different to a line which reaches upwards or sideways from an axis towards a point.
* **Coordinate systems**: these define how space is represented in an image. For example, a Cartesian coordinate system represents space in terms of latitude and longitude, whereas a Polar coordinate system represents space in terms of distance from origin and angle. Many visualisations that look very different are in fact the same, but for the coordinate system used.
* **Aesthetics**: these represent the rules by which different variables are mapped onto different geometric features placed in different positions within the space as defined by the coordinate system(!). This is an initially confusing idea, but in many ways the most important take-home message from the approach. The difference between data visualisation and other forms of information visualisation can mainly be understood in terms of this mapping process.
* **Facets**: these are tabular arrangements of multiple graphs. They are a vital tool for data visualisation meant for internal use (i.e. for exploratory data analysis), and are a simple but very important tool for high information density data visualisation more generally.
* **Guides**: these are additional pieces of information which help the viewer to understand the context of the data visualisation. Examples of guides include legends for distinguishing between different types of line and point, and scales for helping people work back from a position or other aesthetic feature of a graph and the underlying values.

# The Layers of Graphics

There are essentially three layers of a data visualisation:

* The **support layer**, containing background features such as gridlines, tickmarks, axis labels, legends and so on. In terms of the grammatical elements listed earlier, the support layer tends to contain **scales** and **guides**.
* The **data layer**, containing the results of consistently applying a series of rules for mapping from data variables to aesthetic forms. In order to be a data visualisation, a graphic must contain at least one data layer, and at its most minimal a data visualisation can contain nothing but the data layer. The grammatical elements listed earlier that data layers are made up of tends to include **geometries**, **aesthetics** and **facets**.
* The **annotation layer**, containingeverything else. The annotation layer can be what makes an otherwise generic graph stand out, for better or for worse. The annotation layer actively discusses and describes the data within the same frame within which it is being presented. Think of the annotation layer as providing, in print, the added value (and value judgements) that you would provide to an audience if presenting the visualisation in person. Just as you might want to direct the audience’s attention to a particular point or line with a laser pointer, so you might want to add an arrow and some additional text to discuss the data within the visualisation itself.

# 

# The Three Hats of Data Visualisation

To be effective and efficient at data visualisation, we need to learn to swap between three very different frames of mind, three ‘hats’. We need to swap between the **Artist**’s hat, the **Scientist’**s hat, and the **Engineer**’s hat.

* As an **Artist**, you will be concerned with beauty, with proportion, with form, with shape, with engagement. You will approach the task of producing a data visualisation as an opportunity to engage and inspire audiences, to bring feeling to numbers, to express yourself, to make art.
* As a **Scientist**, you will be concerned with accuracy, with precision, with efficiency, with testing and exploring theories, with disentangling and teasing apart complex causal relationships between variables, with learning something new about the world, and explaining what you have found to others clearly and honestly.
* As an **Engineer**, you will have to implement the plans agreed upon by the **Artist** and the **Scientist**, turning sketches and aspirations into reality. You will be faced with overcoming constant, annoying, niggling technical challenges and obstacles to realise the blueprints. You will spend your time answering the question how. How do I access that data? How to I derive this variable? How do I change the line width? How do I label that point? How do I change the tick-mark on the axis? How do I do this? How can I do that?

As an **Engineer**, you will have a raw deal. If you fail to figure out how to do something, you will be blamed for your lack of knowledge and relevant skills. If you succeed, however, you will not be rewarded. Instead, the **Artist** and the **Scientist** will be thanked for the inspiration, boldness and insight of their plan. You merely carried out their instructions correctly, as you are expected to.

However – and this point is critical for understanding what data visualisation is in practice – *you will spend much more time wearing an* ***Engineer****’s hat than an* ***Artist****’s hat or a* ***Scientist****’s hat*. As a rule of thumb: data visualisation is about 80% engineering, 10% art, and 10% science.

# 

# Table Management

As most of the job of data visualisation involves wearing the engineer’s hat, moving from one technical challenge to the next, you can usually improve the effectiveness and efficiency as a data visualiser by learning to be just a little bit more efficient and effective in data engineering issues. An important source of ideas and advice on effective data management comes from database theories, and thinking carefully about the distinction between **keys**, **variables**, and **values**. Crudely, if **keys** show *something that was* measured, **variables** record *what* was measured (of the ‘*something that was measured’)*, and **values** report *the result* of measuring (what was measured of the something that was measured).

As a simple example, the following table can be decomposed into a format where these three fundamental components are made explicit rather than implicit.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Surname** | **First name** | **Month of trial** | **Height** | **Condition** | **Weight** |
| Smith | Mary | 1 | 78 | Healthy | 45 |
| Smith | John | 2 | 62 | Sick | 52 |
| Jones | William | 1 |  | Healthy |  |
| Jones | William | 2 | 86 |  | 84 |
| Smith | Mary | 3 | 78 | Sick | 43 |

Table 1 Example 'Messy' wide format data

|  |  |  |
| --- | --- | --- |
| **Key** | **Variable** | **Value** |
| Smith\_Mary\_1 | Height | 78 |
| Smith\_Mary\_1 | Condition | Healthy |
| Smith\_Mary\_1 | Weight | 45 |
| Smith\_John\_2 | Height | 62 |
| Smith\_John\_2 | Condition | Sick |
| Smith\_John\_2 | Weight | 52 |
| Jones\_William\_1 | Condition | Healthy |
| Jones\_William\_2 | Height | 86 |
| Jones\_William\_2 | Weight | 84 |
| Smith\_Mary\_3 | Height | 78 |
| Smith\_Mary\_3 | Condition | Sick |
| Smith\_Mary\_3 | Weight | 43 |

Table 2 Long format data created by 'melting' Table 1 into its component parts.

Although the long format data is not as human readable, it is often easier to work with and reshape to produce other summary tables and graphs. Time invested in learning more about the fundamentals of data management, including using filters and pivot-tables, merging and   
  
  
  
summarising data, and reading and saving data in a variety of formats (including plain text formats such as .csv), will pay dividends in your work as a data visualiser.

# 6. References

## 6.1 Books

If you only want to read one book on data visualisation, and want to focus on using Excel to produce something professional and consistent, then I would recommend the following book by Steven Few. One of the few ‘how-to’ books that would also look good on a coffee table:

* Few, S. (2012) **Show Me the Numbers: Designing Tables and Graphs to Enlighten.** Second Edition, Analytics Press: Burlington

Other books by Steven Few include:

* Few, S. (2006) **Information Dashboard Design: The Effective Visual Communication of Data.** O’Reilly: Sebastopol, California
* Few, S. (2009) **Now you see it: Simple Visualisation Techniques for Quantitative Analysis.** Analytics Press: Burlington

If you would like to learn more about Edward Tufte’s data visualisation design principles and philosophy, then his books are a pleasure to read and view. The first and most famous of which is:

* Tufte, E. (2001) **The Visual Display of Quantitative Information** Second Edition, Graphics Press: Cheshire, Conneticut

Two books from the ‘infographics’ end of the data visualisation spectrum, which have arguably done much to bring the idea of data visualisation to public consciousness, while also often misrepresenting the field in ways that cause more technically minded data visualisers to wince, are:

* McCandless, D. (2012) **Information is Beautiful.** New Edition, HarperCollins: London (More so)
* Rogers, S. (2013) **Facts are Sacred.** Faber & Faber: London (Less so)

(Declaration: I own both books.)

A more nuanced and balanced book from the world of data journalism is the following:

* Cairo, A. (2013) **The Functional Art: An introduction to information graphics and visualization**. New Riders: Berkeley, California

If you only want to buy two books, I would recommend buying Cairo’s after buying Few’s! Whereas Few’s book will help produce sturdy, clean, simple, honest graphics for helping organisations understand themselves better, **Cairo’s book is an engaging and wide ranging** tour popular data visualisation and infographics. A take-home message from the book is that the general public can probably cope with more information rich visualisations than they’re usually provided with.

If you are feeling brave, and want to learn about how to use statistical programming languages such as Javascript, Python and R to produce data visualisations, then I would recommend the following books by Nathan Yau, a US-based statistician who runs the website flowingdata.com

* Yau, N (2011) **Visualize This: The FlowingData Guide to Design, Visualization and Statistics.** Wiley: Indianapolis, Indiana
* Yau, N (2013) **Data Points: Visualization That Means Something.** John Wiley & Sons: Indianapolis, Indiana

Although Yau’s books are fairly technical, and producing visualisations as he does requires a lot of specialist knowledge in programming, his graphics also demonstrate a high level of care and attention to issues of aesthetics and graphic design. As an effective, all-round data visualiser, there are very few competitors to Yau.

The UK-based data visualisation expert Andy Kirk has written the following book:

* Kirk, A. (2012) **Data Visualization: a successful design process**. Pakt Publishing: Birmingham

It is a book of two halves: the first (shorter) half discusses the process of arriving at the correct design choices for a graphical visualisation; the second half is a reference guide describing and providing a wide range of example of different types of visualisation, and suggestions about the applications for which they would be best suited. I see it as a ‘conceptual’ practical guide: something to help the Scientist and Artist work well together in drafting something suitable for the Engineer to produce, rather than something that directly makes the Engineering involved any easier.

If you would like an even more comprehensive reference of data visualisations, then I would recommend the following:

* Harris, R (1999) **Information Graphics: A Comprehensive Illustrated Reference.** Oxford University Press: New York

Much more a catalogue than a coffee table book. Nonetheless, a good book to skim through if you don’t want to rely on Excel’s default options to spark your imagination.

### *6.1.2 Graphics in R using ggplot2 and lattice*

If you want to continue to use ggplot2, the main book I would recommend, and the book I have been using throughout the workshop is:

* Chang, W. (2013) **R Graphics Cookbook**. O’Reilly: Sebastopol

Together with the various websites and mailing lists for ggplot2, for day-to-day tasks, Hadley Wickham recommends Chang’s book over his own:

* Wickham, H. (2009) **ggplot2** Springer: London

However, if you want to learn more about the mechanics of ggplot2, then Wickham’s book is required reading.

Within the demographic visualisation portion of the workshop, I used an alternative to ggplot2, called **lattice**. Lattice is a somewhat older package to ggplot2, whose popularity has waned as ggplot2’s has waxed. It has a greater level of flexibility and range of features than ggplot2, although ggplot2 is expanding rapidly. However, lattice is much less intuitive to use, and has much less aesthetically pleasing default options. It was created, and is still being developed, by Deepayan Sarkar, whose book on the package is:

* Sarkar, D. (2008) **Lattice**. Springer: New York

Both Lattice and ggplot2 are based on different graphical paradigms. Lattice is based on the ‘trellis graphics’ routines developed by William Cleveland and colleagues at Bell Labs in the early 1990s, for S-Plus, the commercial predecessor to R. Trellis graphics are closely related to the facet feature of ggplot2, although conceived of and specified quite differently.

In contrast, ggplot2 is based on the grammar of graphics paradigm defined by Leland Wilkinson:

* Wilkinson, L. (2005) **The Grammar of Graphics.** Second Edition, Springer: Chicago

In terms of programming mechanics, both lattice and ggplot2 are high level graphical languages built using the same low level graphics language, an R packages known as **grid**. Grid allows very fine-grained control of graphical outputs, but at the cost of requiring a lot of programming expertise and knowledge of graphical devices in R. It was created by Paul Murrell, whose book on R graphics describes grid in great detail:

* Murrell, P. (2011) R Graphics. Second Edition, Chapman: Bosa Roca

If you wish to learn about R more generally, I would recommend the following:

* Kabacoff, R. (2011) **R in Action.** Manning: New York
* Fox, J & Weisberg H. (2011) **An R Companion to Applied Regression.** Second Edition, Sage: London
* Matloff, N. (2011) **The Art of R Programming.** No Starch Press: San Francisco

If you wish to explore R for teaching statistics, I would recommend Andy Field’s new book:

* Field, A., Miles J & Field, Z. (2012) **Discovering Statistics using R.** Sage: London

## 6.2 Websites

When I’m unsure how to do something in R, the first place I look is usually the Quick-R website

<http://www.statmethods.net/>

Ggplot2 has its own website, containing hundreds of examples and vignettes of ggplot2 features:

<http://docs.ggplot2.org/current/>

There is also a ggplot2 mailing list, which Wickham promises is much friendlier than the basic R mailing list:

<http://ggplot2.org/>

If I’m really unsure of how to do something in R, the programming chat room **stackoverflow** is here to help:

<http://stackoverflow.com/>

However, if asking a question on this website, it is important to thoroughly check the website archives, to see if the question has been asked. Also be sure to include a fully reproducible example (i.e. code) of the problem you are trying to solve.

## 6.3 Software

R is open source, and this is both its strength and its weakness. It is a strength because it has encouraged thousands of packages to be developed; for example, almost as soon as a statistician has developed a new model, an R package implementing that model will follow. However this is also a weakness as it leads to inconsistency and variation in the syntax and structure adopted by different packages, as each package is the product of a different thought process. This means that R can feel very incoherent when compared with a standard statistical software package: using a new package often means learning a new dialect, a new set of terms, a new way of specifying and instructing. R, as a language, has the linguistic variation of Europe, with its many languages and regional dialects, rather than the United States.

However, Hadley Wickham, the creator of ggplot2, has in recent years done much to establish a coherent, consistent and clear *lingua franca* within R. In addition to ggplot2, he has developed a number of packages for data management and related tasks, all of which work well together, and follow a common underlying design philosophy. These packages include **stringr**, for working with character strings, **reshape2**, for reshaping datasets, and more importantly **plyr**. Plyr applies what Wickham terms the ‘split-apply-combine’ paradigm to data analysis. (See <http://www.jstatsoft.org/v40/i01/paper>) The main aim of these packages is to allow the R user to focus more on what they want to achieve conceptually, and less on how to write the code to achieve this in R. In short, they help R to feel more like other statistical packages. If you wish to learn R for the first time, I would recommend installing Wickham’s packages to have it speak in the Wickham dialect.

Although the focus throughout the course has been on the R language, the data visualisation principles are more generic. I often use Excel, for all its faults, when first exploring some simple datasets. In some cases, and when producing something as a ‘one-off’, Excel can be coerced into producing a decent visualisation faster than R. There’s no shame in resorting to Excel (I tell myself). Just recognise that Excel has bad taste in terms of graphical defaults and aesthetics, and so graphics produced in Excel are likely to require some tweaking. There is a template option in Excel, meaning that once you’ve produced a graph with good graphical default options, you can use it again and again.

In terms of the three graphical layers – support, data, and annotation – R is perhaps least appropriate for the annotation layer. Although there are options within ggplot2 to add labels, to highlight particular aspects of a display, and so on, there are likely to require learning a lot of programming to do something comparatively small. Nathan Yau, creator of the website FlowingData, recommends using R to produce a first iteration of a graphic, then polishing the aesthetics and adding annotations using **Adobe Illustrator,** the main tool used by graphical designers. Examples of this two-stage process are described in his books, referenced earlier. Illustrator is fairly expensive; an open source alternative is **InkScape**. However, Illustrator is very highly regarded, and considered well worth the expense by those who repeatedly need to produce engaging visualisations for the general public.

If you like the principles of the grammar of graphics, but feel overwhelmed and underwhelmed by R, then there is a commercial package, called **Tableau**, which offers many of the same features, is much easier to use, and has even more aesthetically attuned graphical default options. Conceptually, try to imagine the pivot table and pivot chart options in Excel, but *much, much better*, with better handling both of data management and visual design. Tableau comes in both a public version, and a commercial version. The difference between the versions is not in the functionality, but in the storage of the data. In the public version, data has to be uploaded to Tableau’s website in order to be used. In the commercial version, you can keep the data to yourself. The cost of secrecy is about £1,000. The public version of Tableau can help you decide whether, for your organisation, this is a price worth paying: <http://www.tableausoftware.com/>