 Dynamic report demo

2022-06-10 11:04:31

Table of Contents

# Introduction

This is an example of a dynamic report, written in [R](https://www.r-project.org/) and [Rmarkdown](http://rmarkdown.rstudio.com). It gives some example ways of automating common data-driven tasks. This report is intended as a starting point for automating your report writing processes. The idea is to replace (or speed up) manual report writing. The demo gives some basic introduction to the workflow. It then walks through a series of example tasks based on a toy data set. The demo also covers ways of producing the report in a variety of formats including pdf, Word document, and html webpage.

## Who is this demo for?

This demo is intended for someone who:

* is currently spending lots of work time updating routine reports in health, care, or housing
* **and** would like to reduce the time spent on this routine work over the medium-term (months)
* **and** currently have the time to spend doing a complete overhaul of your report(s)
* **and** with a bit of coding experience **or** open to gaining some as part of this process
* **and** with an hour or two now to work through this demo

## What this demo is not

* it’s not a full introduction to working in R/Rmarkdown. R is deep (like most analytic platforms), and this demonstration barely scratches the surface
* it’s not an introduction to everything that you might like to do with a data-driven report either. Instead, it covers some simple examples of common tasks as a way of familiarising you with a different workflow
* it’s definitely not a production-ready replacement for your existing reports. Instead, it’s a learning resource, rather than a pre-packed replacement

## Is it worth it for me?

That’s a metaphysical question!

* Much harder the first time, and slower
* So if you have regular reports, especially complicated ones with lots of graphs, deffo
* Local IG concerns

## How does it work

[R](https://www.r-project.org/) is a programming language designed for statistical work

[Rmarkdown](http://rmarkdown.rstudio.com) is a simple markup language

Closely related to other kinds of markdown - you might well have used this before without really knowing what it was called. If you’ve ever posted on a wordpress site or web forum with basic text formatting options available, the chances are that you’ve already done at least a tiny bit of markdown. Just to give you an example, surrounding a word with asterisks formats it in *italics*. Or putting a dash before a word:

* puts it as a list item

This kind of very basic formatting is Markdown: so Rmarkdown is a markdown with a few extras to allow you to embed R code directly into the text that you’re writing.

You might like to have a quick look at some of excellent introductory R resources **add more resources**

* [Rstudio’s quick tour of Rmarkdown](https://rmarkdown.rstudio.com/authoring_quick_tour.html) is a great place to start if this is all new to you
* If you’re already somewhat familiar with working in R, you might prefer to start with the [Rmarkdown chapter in the *R for Data Science* book](https://r4ds.had.co.nz/r-markdown.html)

## What you need to work through the demo

* working R and Rstudio installation **or**
* access to a RStudio Cloud account

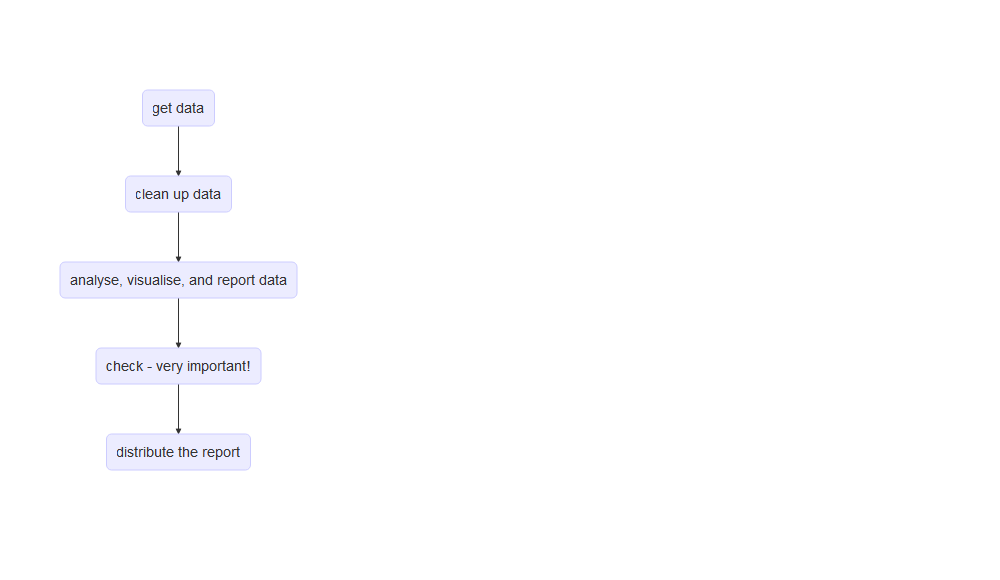
Easy to set-up and free for small-scale work. Likely not suitable for production work in health and care owing to information governance concerns, but by far the easiest way to get going from scratch if you’ve never worked with R before.

* the source code for this report - [GitHub link](https://raw.githubusercontent.com/bclarke-nes/Dynamic-report-demo/master/demo.Rmd)
* the demo data file [demo.csv](https://raw.githubusercontent.com/bclarke-nes/Dynamic-report-demo/master/demo.csv)
* some experience of building reports manually
* a good idea of what you’d like your report to do
* time and space to complete the demo, and think about how it might be useful in your own practice

My suggestion would be not to try and use this demo to change your way of writing reports under pressure. There’s quite a lot to think about here, and you might need to spend a good bit of time working out how to adapt this demonstration to fit your report. Think of this as the start of a journey, rather than a destination.

## Workflow overview

This demonstration will take you, step by step, through building a simple report in R and Rmarkdown. The big difference here from the usual way of building reports by hand is that you will combined the analysis, visualisation, and production stages of your report into a single step. As this demo will show, you can do all of this using R and Rmarkdown. You’ll load the data, clean and analyse the data, and produce outputs (text, tables, graphs) straight into your report.



# Starting an Rmarkdown document

There’s a bit of preamble at the head of Rmarkdown documents that controls how they are built. For now, I’ll suggest that you skip trying to figure it all out, and just use the header from this document. I’ll give more details about setting up your Rmarkdown document at the end of this workthrough, but for now let’s just skips ahead to the interesting part where we get some data into our project.

Once you’ve got the header in place and tweaked to your liking, we’re ready to move on to something more useful: getting data into your report.

# Get data

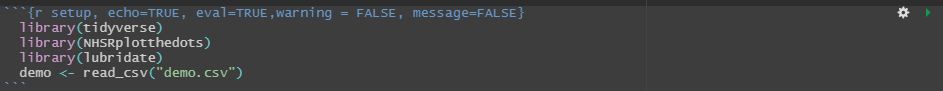
This feelsis easy to do in practice, but is a bit of a conceptual leap if you’re used to collecting and receiving data in Excel. Instead of opening a data file directly with Excel so that you can inspect and analyse it, we instead load all the data in that file into R before we can use it.

Because there are lots of different types of data out there, there are lots of ways of loading data into R.

Let’s look at the first R chunk of our doc:

library(tidyverse)  
 library(NHSRplotthedots)  
 library(lubridate)  
 demo <- read\_csv("demo.csv")

Code chunks in Rmarkdown start and end with a triple backtick:



There are then a few options that you can set for the code chunk. Here:

* r: telling Rmarkdown to interpret this code as R. There are other languages that you can use in Rmarkdown, which is one of the strengths of building reports in this way.
* setup: the chunk label. This can be anything you like, but no duplicates are allowed. Very useful for navigation in more complicated reports
* echo=TRUE: whether to show this code in the report. If this is set to FALSE, you’ll just see the output of the code (more on this below)
* eval=TRUE: whether or not to run the code. If eval=FALSE, the code won’t do anything other than appear in the report (useful if you’re explaining how something works - like this report!)
* warning = FALSE: whether to show warnings in the report if something goes wrong with your code
* message=FALSE: whether to show information messages about how your code is running

Lots more info in the [Rmarkdown Cookbook](https://bookdown.org/yihui/rmarkdown/r-code.html)

That chunk of R code is doing two different things: + the three lines starting library(...) load three packages that help you do your analysis. More info

you can also use knitr::opts\_chunk$set, which is a really powerful way of managing chunk options - but that’s a bit beyond the scope of this demo.

# Clean up data

Time for another R chunk:

demo <- demo %>% mutate(date = dmy(date))

# Getting text to update to reflect data

Let’s have a look at the what’s in demo.csv. We load this into the report in the setup chunk, using

demo <- read\_csv("demo.csv") %>% mutate(date = dmy(date))

That gives us a tibble as follows:

| date | count1 | count2 | count3 |
| --- | --- | --- | --- |
| 2022-05-01 | 11 | 93 | 82 |
| 2022-05-02 | 15 | 114 | 99 |
| 2022-05-03 | 18 | 28 | 10 |
| 2022-05-04 | 19 | 107 | 88 |
| 2022-05-05 | 21 | 140 | 119 |
| 2022-05-06 | 170 | 217 | 47 |

(there are another 62 rows that I’ve cut out to save space).

This data will change as the data in the csv file changes. I’ll encourage you to fiddle about with this, but if you’re used to copy and pasting data tables about to update, this will be a great help.

We can also produce summary text inline. For example, we can count up all of the rows here, and say that we have a total of 68 records. Or we can add all the entries in the count1 column (a total of 6643). Or find out that the maximum value in count3 is 141. Or we can say that on the first day of this range (2022-05-01), the value of count2 was 93. Or that the mean (average) of count3 was 70.5.

I’ll say it again: **change the data, and all this stuff will update automatically**. No need to copy and paste at many locations. For example, you might like to include a date in your report describing which month the data is about. You can do this easily: the latest date in this data comes from July. Change the data, and this report will update - go on, try it!

# Doing data processing

We can do useful stuff like add the columns together:

| date | count1 | count2 | count3 | total |
| --- | --- | --- | --- | --- |
| 2022-05-01 | 11 | 93 | 82 | 186 |
| 2022-05-02 | 15 | 114 | 99 | 228 |
| 2022-05-03 | 18 | 28 | 10 | 56 |
| 2022-05-04 | 19 | 107 | 88 | 214 |
| 2022-05-05 | 21 | 140 | 119 | 280 |
| 2022-05-06 | 170 | 217 | 47 | 434 |

(again, just showing the first few rows of data).

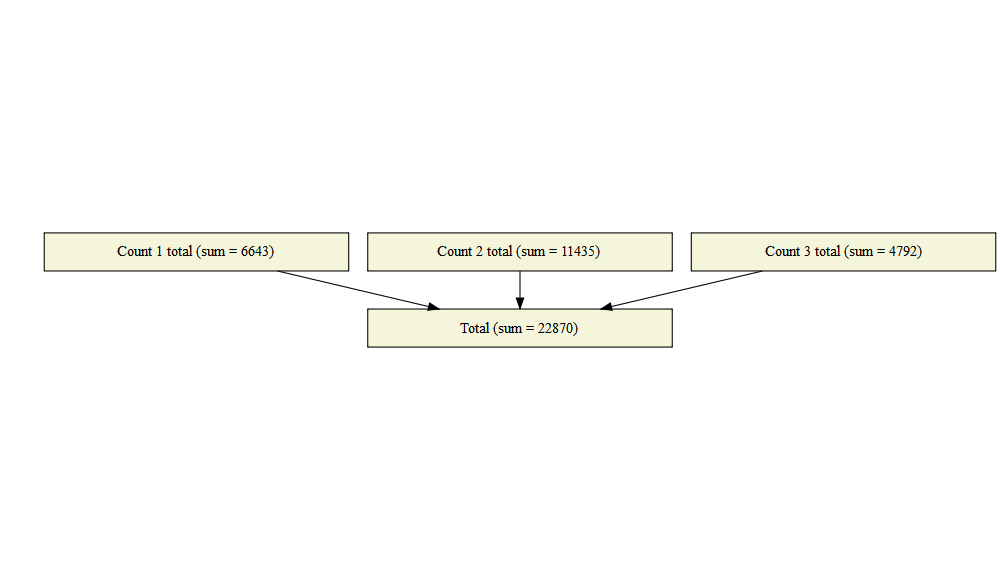
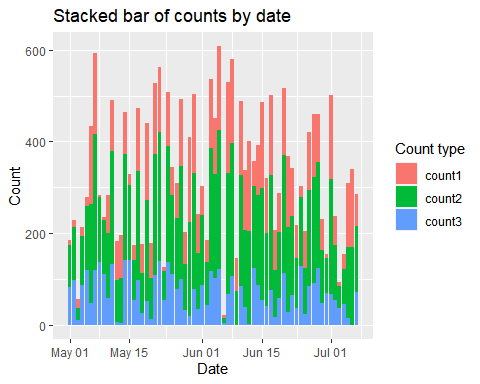
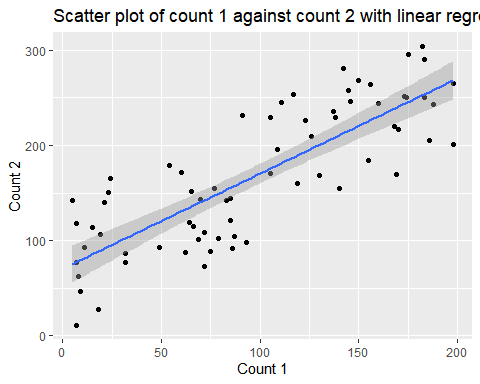
We can also summarise tables of data:

| count1 | count2 | count3 | total |
| --- | --- | --- | --- |
| 98 | 168 | 70 | 336 |

or re-name columns in our tables:

| Mean of count 1 | Mean of count 2 | Mean of count 3 | Mean of total |
| --- | --- | --- | --- |
| 98 | 168 | 70 | 336 |

# Graphs



# Rmarkdown document settings

The easiest option is to start a new Rmarkdown document direct from the RStudio menu File >> New file >> R Markdown.... Alternatively, you can copy the header from this document, and paste it at the top of your document. Everything between the pair of three dashes: --- is concerned with setting up this Rmarkdown document:

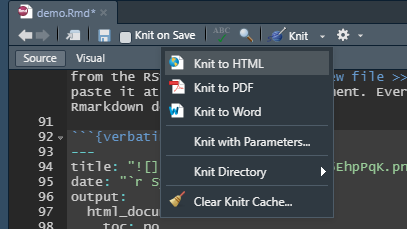
---  
title: "![](https://i.imgur.com/5EhpPqK.png) Dynamic report demo"  
date: "`r Sys.time()`"  
output:  
 html\_document:  
 toc: no  
 toc\_depth: 2  
 number\_sections: no  
 toc\_float:  
 collapsed: no  
 pdf\_document:  
 toc: yes  
 toc\_depth: '2'  
 pandoc\_args: ["--extract-media", "."]  
 word\_document:  
 toc: yes  
 toc\_depth: '2'  
---

You might notice that there are a couple of differences between the header from this document and the default version that RStudio gives you. I’ve added a link to the KIND Learning Network banner image as part of the title. If you’re thinking about adapting this report to suit your purposes, you can easily change both of those by editing the URL (and the title text) in the following line:

title: "![](https://i.imgur.com/5EhpPqK.png) Dynamic report demo"

More info on [YAML editing in chapter 2 of the excellent R Markdown Cookbook](https://bookdown.org/yihui/rmarkdown-cookbook/rmarkdown-anatomy.html).

Most of this header block is made up of options for the different output formats (html, word, and pdf). In RStudio, you can select which output format you’d like your report to be rendered as:

 Try these out now - this demo report should knit to each of the three formats without any problems. I’d recommend sticking with html while you’re working on a report, though, just because it tends to render most quickly, and plays nicely with other aspects of the workflow while you’re writing.

One thing to watch out for when tweaking these options: indentation and white-space matters. Each of the three output formats (html\_document, pdf\_document, and word\_document) are indented from the *output:* line, and each of the output format options are indented too. A common cause of documents not knitting (see below) are small errors in the indenting in the header.

You might notice another tweak that I’ve made to the pdf output options for this document, which is to add the line:

pandoc\_args: ["--extract-media", "."]

# So what’s next?

## Alternatives

In the R/Rmarkdown universe, [Flexdashboard](https://pkgs.rstudio.com/flexdashboard/) looks great: there’s an [example](https://www.themillerlab.io/post/flexdashboards/mcc_fake_cohort_dashboard_for_post.html#outcomes-by-stage) and a [walkthrough by the Miller Lab](https://www.themillerlab.io/post/flexdashboards/)