From Scripts to Projects: Learning a Modular, Auditable, and Reproducible Workflow by Example

Jessica Randall

12/29/2019

Contents

Introduction

I first learned about Dr. Patrick Ball's work on contributing statsitical evidence to the conviction of Slobodan Milosevic through my fiance and was immediately inspired as I had been starting to take statistics courses in my graduate program at the Rollins School of Public Health at Emory University. Upon mentioning Patrick's work and that he had co-founded HRDAG with Dr. Megan Price to a mentor at Emory (Hi Paul!) I found out what a small world it was and my own mentor had also been one of Megan's. I have had the immense pleasure to chat with both Megan and Patrick over the last year about their work and I was thrilled when I received an email from Patrick about a month ago asking if I would be willing to assist on an HRDAG project.

I had read about the organization's modular workflow (link) and watched Patrick's talk on principled data processing (link) but as someone who had only formally learned SAS for a couple of years and dabbled in self-taught Python and R for a handful of months, it was initially intimidating. I fully admit to my R code looking very much like the examples of amateur coding that Tarak used in his recent post on why .Rproj is harmful to reproducible and auditable data science (link).

Eager to jump in to a crash course in the HRDAG process necessitated by the client's need for an immediate answer, I had many a late night cursing Hadley Wickham (sorry, Hadley, you're awesome and so is the tidyverse) but truly enjoyed being able to contribute even just a tiny bit to HRDAG's amazing work.

With the import of what our answer could mean for our client's activism in mind and with a great deal of Patrick's guidance I shifted towards working mainly in the command line as opposed to R Studio, learning Vim, and to generally start thinking of scripts in terms of a larger project architecture as opposed to one-off scripts I would write per project.

As I worked, Patrick also encouraged me to keep notes on what I liked and disliked about the process, where I felt it made things easier and where I noticed friction. The following post summarizes those notes and the work that went into determining the answers to the client's question.

At the risk of falling into the caveat fallacy, I first want to clarify that we assume these data are true and that they represent a sample of reported deaths rather than a complete record of all actual deaths during this time period. Any conclusions we come to as a result of our analyses are contingent on these data being true and come with the caveat that this sample is likely underreported and we don't know what we don't know.

To begin the project, we start with a task. Each task is a microchasm of the entire project and at HGRAG this facilitates the team's collaboration across time zones and programming languages. It also results in projects which are auditable and reproducible. Anyone looking at the code can determine exactly how any output from it was produced and anyone working on the task at anytime can pick it up and recreate the same output you wrote. This need for reproducibility resonated with me and even if you are a team of one in your workplace (as many statisticians reading this might be) your closest collaborator is you six months from now. Break up your project into discrete tasks and thank yourself later.

To make the goals of each task even clearer, each task has a Makefile. The Makefile serves as an outline of each task not only for our benefit, but mainly for the computer to read it. Imagine being able to work on a group project with team members around the world, and still have an auditable trail of each person's work. The Makefile makes that scaffolding clear and traceable. For more information on Makefiles check out Mike Bostock's post here (link) and for the inspiration for HRDAG's project oriented workflow check out Jenny Bryan's post here (link).

For this project I have a clean task, a test task, and a write task each with its own Makefile, input, output, and src folder for the source code.

Clean

I start with the clean task. In the input folder of this task I include the two datasets of with reported deaths, the src folder contains the R code to clean up the code. I used RStudio to protype new ideas for how to code something, test the code for bugs (with frequent restarts and fresh environments) and use Microsoft Visual Studio Code with a Vim extension to update the code that gets pushed to the project Github repository. I had a small bug early on with an error message I didn't want to immediately address so I created an issue on the GitHub to pick it back up later. I knew where the code had issues but I could count on the rest of it working as expected if (when) I'd need to spend time on other projects. This also had the added benefit (trap?) of making me feel like I'd actually made progress instead of staring down the endless empty field of .R file with however many tasks left to code. Win!

The first dataset includes initial deaths reported from 12 June 2019 – 10 September 2019. The second file contains additional deaths reported from 1 January 2019 – 14 December 2019. Our client would like us to determine if the number of deaths reported during a specific time period are unusually high following a specific date of interest.

To clean the data, we make sure that we don't have strings of text where we expect to see numbers and vice versa using informal unit tests. This means that instead of running the code and interactively checking the dimensions or the top 5 rows of a dataset visually, we use the assertr package to specify some condition the data has to meet and tell our code to stop running if that condition is not met.

We also canonicalize the data. This refers to ensuring that the names of the variables we are comparing are standard across the datasets if we were to combine them. We are working with two datasets here and since they overlkap in terms of when they were collected, we'd like to keep them seperate while still having variables named consistently.

Since we are interested in examining only those cases in which people have been reported as dead, we remove all rows where the person's status is "Not_death". We are also interested in reported deaths occuring prior to or following a specific date so we created a variable indicating whether or not a death occured prior to or following that date.

The data are cleaned as part of the clean task and all of that code is stored in the clean.R file in the clean/src folder. To use the clean data, we pull it from the clean task's output folder.

```
files <- list(input1=here::here("clean/output/death1_clean.txt"),
              input2=here::here("clean/output/death2_clean.txt"))

death1 <- readr::read_delim(files$input1, delim="|")

death2 <- readr::read_delim(files$input2, delim="|")

head(death1)
```

```
## # A tibble: 6 x 4
##   date       status DOD        dateb_20190821
##   <date>     <chr>  <date>     <chr>
## 1 2019-06-12 dead   2019-06-12 pre
## 2 2019-06-13 dead   2019-06-13 pre
## 3 2019-06-15 dead   2019-06-15 pre
## 4 2019-06-15 dead   2019-06-15 pre
## 5 2019-06-15 dead   2019-06-15 pre
## 6 2019-06-15 dead   2019-06-15 pre
```

```
head(death2)
```

```
## # A tibble: 6 x 4
##   date       status DOD        dateb_20190821
##   <date>     <chr>  <date>     <chr>
## 1 2019-12-14 dead   2019-12-14 post
## 2 2019-12-14 dead   2019-12-14 post
## 3 2019-12-13 dead   2019-12-13 post
## 4 2019-12-13 dead   2019-12-13 post
## 5 2019-12-13 dead   2019-12-13 post
## 6 2019-12-12 dead   2019-12-12 post
```

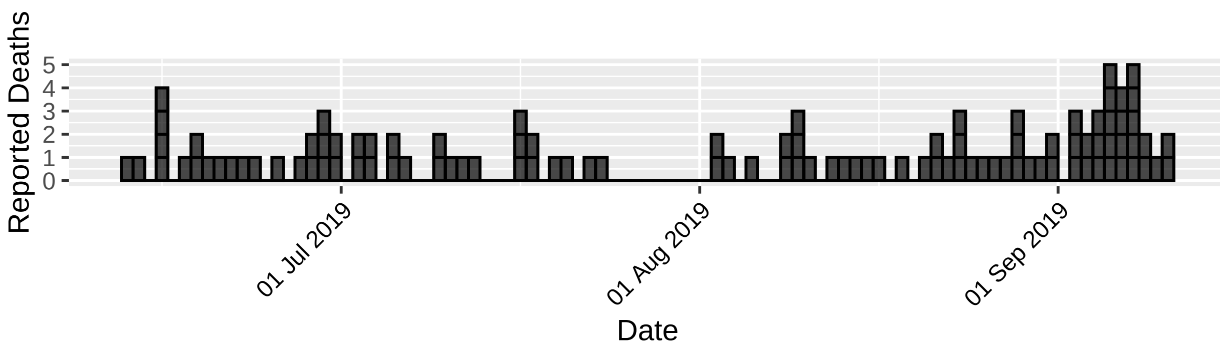
Epi Curve 1

In order to see if there appear to be an abnormally high number of deaths, we will first use an epi curve. Epi curves are used in investigations of potential outbreaks and their shape can even suggest a means of propogation; curves for infections transmitted from person to person have different shapes than curves of diseases transmitted from a water source, for example. In our data, if an epidemic of deaths were occuring, we would expect to see a spike in the number of deaths reported per day.

The first dataset includes deaths reported from 12 June 2019 – 10 September 2019, 91 days or approximately 3 months. The second set contains deaths reported from 1 January 2019 – 14 December 2019, 348 days or approximately 11 months.

In order to examine the frequency of reported deaths per day, we use the incidence package and plot this against the dates in the dataset.

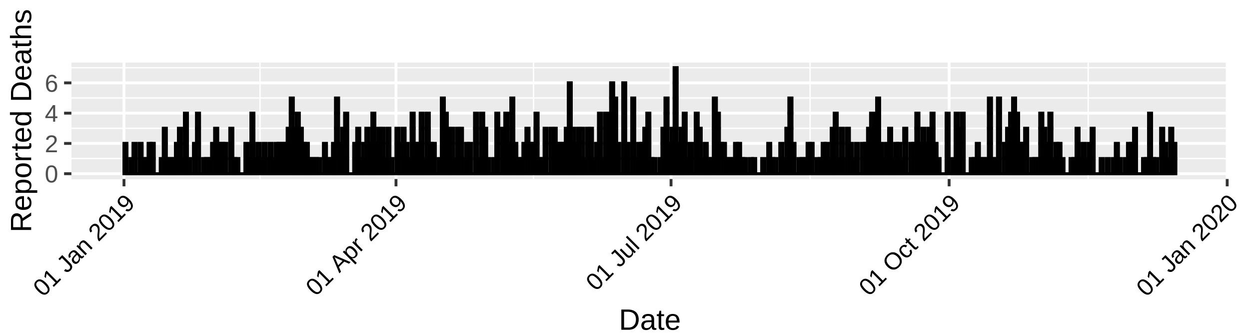
Since these graphs require relatively short blocks of code and don't require major changes to the data, I did not make them a seperate task.



This pattern suggests reported deaths per day increased in the beginning of June and end of August into September. However, with such a small group of people in such a small amount of time, it is not clear that these numbers are abnormally high since we don't know what normal looks like for this population.

Let's examine the 2nd, larger set of data over a longer time period.

Epi Curve 2



Now we have big(ger) data and this graph suggests that reported deaths per day increased in June or July. However, with more data it appears even less clear if the days which have higher numbers of deaths are outside of the normal variation for this population.

I found it humbling and harrowing to see these graphs knowing that they represented individual people whose deaths may or may not have been related to important events. I felt the moral responsibility to each person and I wanted to do the best work technically possible to honor the work being done by our client to investigate these deaths. I felt extremely privileged to work on this project and breaking down each part of it into modular tasks helped me compartmentalize and focus. No matter what your area of research is, there might be times when being able to walk away for a bit and pick up right where you left off knowing all of your work is exactly as it was is a huge relief.

With data in the human rights context it can be difficult to obtain reliable data of a particular outcome of interest, especially if some powerful group has an interest in that data not existing or it was simply never collected. Since we do not have any data on the underlying rates of actual deaths in this population, we tried a more classical statistical approach.

Test

These reported deaths are counts collected in a specific time period. This means that they can be assessed in the context of whether or not they represent a normally distributed discrete random variable in a Poisson distribution. Data which follow a normal Posisson distribution exhibit expected amounts of variation around the mean number of events in a given time period.

We will test the null hypothesis that the mean number of reported deaths does not differ significantly following a date of interest. Our client would then, hopefully, be able to use our findings as a tiny footnote in the larger context of their work.

To test this hypotheses, we call in data from the aptly named test task. In this task we prepared our data by obtaining the frequency of deaths reported per day. Since at least one death wasn't reported at least once per day we also added in dates where we had no reported deaths with frequencies of zero so we don't bias the mean away from the null.

We use the poisson.test function in R's stats package to determine if the differences in the means prior to and following our date of interest, 21 August 2019, are significantly different from one another at the 95% confidence interval. Since we have reason to believe that the frequency of deaths reported per day would increase rather than decrease following this date, we use a one-sided test of the upper limit of the probability distribution. If we observe a p-value < 0.05, we would reject the null. A rejection of the null suggests that the observed mean frequency of reported deaths following the date of interest is abnormally high.

```r
files <- list(cts1 = here::here("test/output/counts1_pre.txt"),
              cts2 = here::here("test/output/counts1_post.txt"),
              cts3 = here::here ("test/output/counts2_pre.txt"),
              cts4 = here::here ("test/output/counts2_post.txt"))

cts1pre <- readr::read_delim(files$cts1, delim="|")
cts1post <- readr::read_delim(files$cts2, delim="|")
cts2pre <- readr::read_delim(files$cts3, delim="|")
cts2post <- readr::read_delim(files$cts4, delim="|")

# set 1
# mean deaths per day prior to 21 August 2019
mu1pre <- round(mean(cts1pre$n))

# mean deaths per day following 21 August 2019
mu1post <- round(mean(cts1post$n))

#test
res1 <-poisson.test(mu1pre, 1, mu1post, alternative = "g", conf.level = 0.95)

# set 2
# mean deaths per day prior to 21 August 2019
mu2pre <- round(mean(cts2pre$n))

# mean deaths per day following 21 August 2019
mu2post <- round(mean(cts2post$n))

#test
res2 <- poisson.test(mu2pre, 1, mu2post, alternative = "g", conf.level = 0.95)
```

In the first dataset, prior to 21 August 21 2019, there was an average of 1 death reported per day and after 21 August 2019 there was an average of 2 deaths reported per day. We cannot reject the hypothesis that the mean number of deaths reported after 21 August 2019 is not significantly greater than the mean number of deaths reported prior to 21 August 2019 at the 95% confidence interval ($p = 0.86$).

In the second dataset, prior to 21 August 21 2019, there was an average of 2 deaths reported per day and after 21 August 2019 there was an average of 2 deaths reported per day. We cannot reject the hypothesis that the mean number of deaths reported after 21 August 2019 is not significantly greater than the mean number of deaths reported prior to 21 August 2019 at the 95% confidence interval ($p = 0.60$).

Working on the uncertain assumption that this is a reliable list of deaths, the frequency of deaths reported during this time period is consistent with what we would expect to see with a normally occurring random process, that is, nothing we're seeing from these data indicates anything out of the ordinary. I will admit to being incredibly relieved but also waking up in a cold sweat regularly hoping I hadn't made some terrible mistake and arrived at the wrong conclusion. Bad data analysis is worse than none at all.

Write

Finally, in what struck me as the coolest technical part of this process, is the write task. This entire post was generated from an R markdown file stored in the write task's src folder. This still excites me and now I want to write as many reports (link) and even presentations (link) as I can this way. An especially fun detail is the use of embedded variables. Throughout this post I've included statistics about this dataset. If we suddenly received more data and I had to regenerate this report, instead of having to manually check that each number I cited was updated and typed correctly, I could drop the files into the clean task's import folder and update the entire report with the knowledge that all of those variables would be correct.

Acknowledgements

Collaborating with HRDAG, specifically Dr. Patrick Ball, on this project has been an incredible learning opportunity for me. Even working in public health, as a statistician I am used to having a comfortable amount of psychological distance from the actual people my work is impacting. Rather than seeing this distance in opposition to the ability to do strong work, it is absolutely necessary that I have it to do the best work technically possible. I started learning how to look at projects from the very macro ("How many tasks do I have?") to the very micro ("Can I pipe this line?") levels. I began to learn how to use the distance inherent in the job of looking at data points and the modularity of the workflow to help me stay focused. This all has offered me an opportunity for a fundamental shift in the way I had been working and I know I will take this into my future projects.

The project architecture structure discussed in this post, with discrete tasks using Makefiles, was developed by Drs. Scott Weikart and Jeff Klingner and formalized around 2007. I am incredibly grateful for Tarak Shah's motivating post on the dangers of .Rproj style of project management, for Dr. Amelia Hoover Green's detailed post on what it means to clean and canonicalize data, Dr. Megan Price's encouragement to work in this field, and Dr. Patrick Ball's invitation to collaborate, his patient explanations, and detailed feedback throughout this project.