Workflow

Andy Grogan-Kaylor
University of Michigan

2026-01-16

Table of contents

1 Introduction

I have increasingly been thinking about the idea of *workflow* in data science / data analysis work.

So many workflows follow the same conceptual pattern.

2 Visually and Conceptually

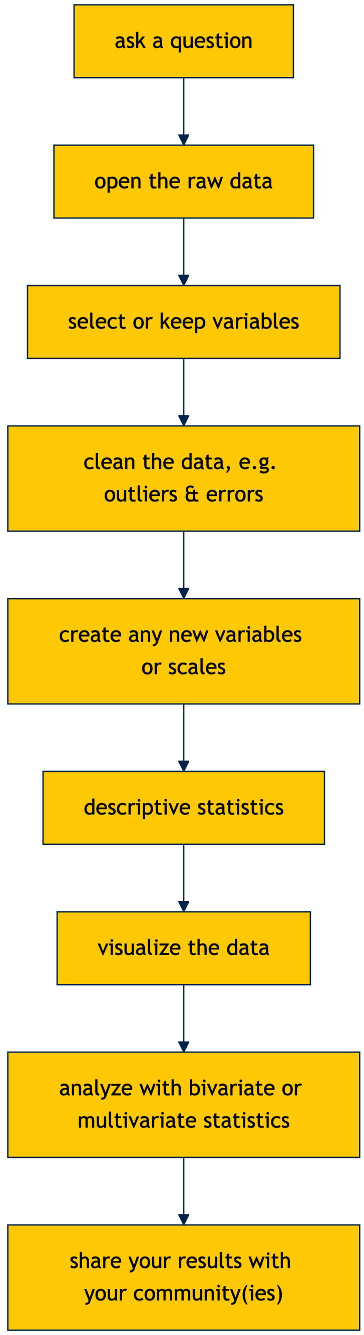

Figure 1: A Common Data Workflow

3 Characteristics of Good Workflows

Increasingly, we want to think about workflows that are

- **documentable**, **transparent**, and **auditable**: We have a record of what we did if we want to double check our work, clarify a result, or develop a new project with a similar process. We, or others, can find the inevitable errors in our work, **and correct them**.
- **replicable**: Others can replicate our findings with the same or new data.
- **scalable**: We are developing a process that can be as easily used with *thousands* or *millions* of rows of data as it can with *ten* rows of data. We are developing a process that can be easily repeated if we are *constantly getting new or updated data*, e.g. getting new data every week, or every month.

4 Complex Workflows

For **complex workflows**, we will often want to write a script or code.

> Complex Workflows Benefit From Scripts or Code
>
> The more graphs or calculations I have to make, the more complex the project, the more the desires of the community or client are likely to change, the more frequently the data is being updated, the more team members that are involved in the workflow, and/or the more mission critical the results (i.e. I need auditability, documentation, and error correction) the more likely I am to use a scripting or coding tool like Stata or R.

Table 1: Tools for Different Workflows

	Simple Process: Single Graph or Calculation	Complex Process: Multiple Graphs or Calculations.
Process Run Only Once	Spreadsheet: Excel or Google	Scripting Tool: Stata or R
Process Run Multiple Times (Perhaps As Data Are Regularly Updated)	Scripting Tool: Stata or R	Scripting Tool: Stata or R

5 Best Practices

> Start With The Raw Data, Do Your Thinking In Code, And Document Your Thinking In Code
>
> Always (or usually) beginning with the raw data, and then writing and running a script or code that generates our results allows us to develop a process that is **documentable**, **auditable**, **replicable** and **scalable**.

Data Are Often Best Stored In Statistical Formats

It is usually best to store quantitative data in a statistical format such as R, Stata, or SPSS. Spreadsheets are likely to be a bad tool for storing quantitative data.

! Good Workflows Require Safe Workspaces

It is also *very important* to be aware that good complex workflows are *highly iterative* and *highly collaborative*, often requiring *a lot of conversation*. Some–hopefully small–amount of error is *unavoidable* and *inevitable*. Good complex workflows require a *safe workspace* in which team members feel free to talk through their ideas, admit their own errors, and help with others' mistakes in a non-judgmental fashion. Such a *safe environment* is necessary to build an environment where the *overall error rate* is low.

! Good Workflows Require Patience And Can Be Psychologically Demanding

Developing a good documented and auditable workflow that is implemented in code requires a lot of patience, and often, **many iterations**. Working through these many iterations can be psychologically demanding. It is important to remember that careful attention to getting the details right early in the research process, while sometimes tiring and frustrating, will pay large dividends later on when the research is reviewed, presented, published and read.

6 Example

Below is an example that uses the Palmer Penguins data set.

This Example is in Stata

The example below is in Stata, due to Stata's ease of readability, but could as easily be written in any other language that has scripting, such as SPSS, SAS, R, or Julia.

```
* Learning About Penguins

* Ask A Question

* What can I learn about penguins?
```

```
* Open The Raw Data
```

```
use "https://github.com/agrogan1/Stata/raw/main/do-files/penguins.dta", clear

* Clean and Wrangle Data

generate big_penguin = body_mass_g > 4000 // create a big penguin variable
```

```
* Descriptive Statistics

use "https://github.com/agrogan1/Stata/raw/main/do-files/penguins.dta", clear

dtable culmen_length_mm culmen_depth_mm flipper_length_mm body_mass_g
i.species
```

```
                        Summary
-------------------------------------
N                                 344
culmen_length_mm       43.922 (5.460)
culmen_depth_mm        17.151 (1.975)
flipper_length_mm    200.915 (14.062)
body_mass_g       4,201.754 (801.955)
species
  Adelie                  152 (44.2%)
  Chinstrap                68 (19.8%)
  Gentoo                  124 (36.0%)
-------------------------------------
```

```
* Visualize The Data

use "https://github.com/agrogan1/Stata/raw/main/do-files/penguins.dta", clear

graph bar body_mass_g, over(species) // bar graph

quietly graph export "mybargraph.png", replace

twoway scatter culmen_length_mm body_mass_g // scatterplot

quietly graph export "myscatterplot.png", replace
```

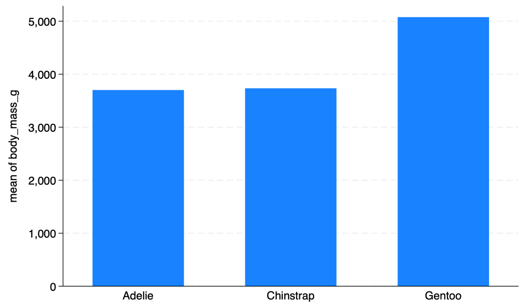

Figure 1: Bar Graph of Penguin Species

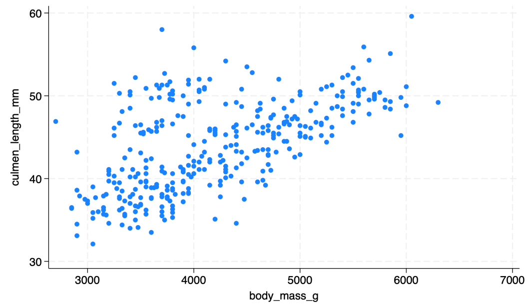

Figure 2: Scatterplot of Culmen Length by Body Mass

```
* Analyze

use "https://github.com/agrogan1/Stata/raw/main/do-files/penguins.dta", clear

quietly: regress culmen_length_mm body_mass_g // regress culmen length on body
mass

estimates store M1 // store these estimates

etable, estimates(M1) showstars showstarsnote // nice table of estimates
```

```
                       culmen_length_mm
---------------------------------------
body_mass_g                  0.004 **
                            (0.000)
Intercept                   26.899 **
                            (1.269)
Number of observations         342
---------------------------------------
** p<.01, * p<.05
```

7 Multiple Person Workflows

When workflows involve multiple people, all of the above considerations apply, but the situation often becomes more complex. Two hypothetical multiple person workflows are illustrated below.

In the diagram below, one workflow is *uncoordinated*. Each person's work is not available to the others, which may cause difficulties if people's work is supposed to build on the work of others. If one team member makes updates or corrects errors, the results of these efforts are not automatically available to the others.

In contrast, in the diagram below, one workflow is *coordinated*. Each person's work is available to the others so that updates and corrections to errors are propagated through the workflow, and into final analyses and visualizations.

It is often the case that a *coordinated* workflow requires more *coordination*, *time*, *energy*, and *patience* to implement than an *uncoordinated* workflow, but a *coordinated* workflow is likely to pay benefits in terms of all of the advantages of good workflows listed above.

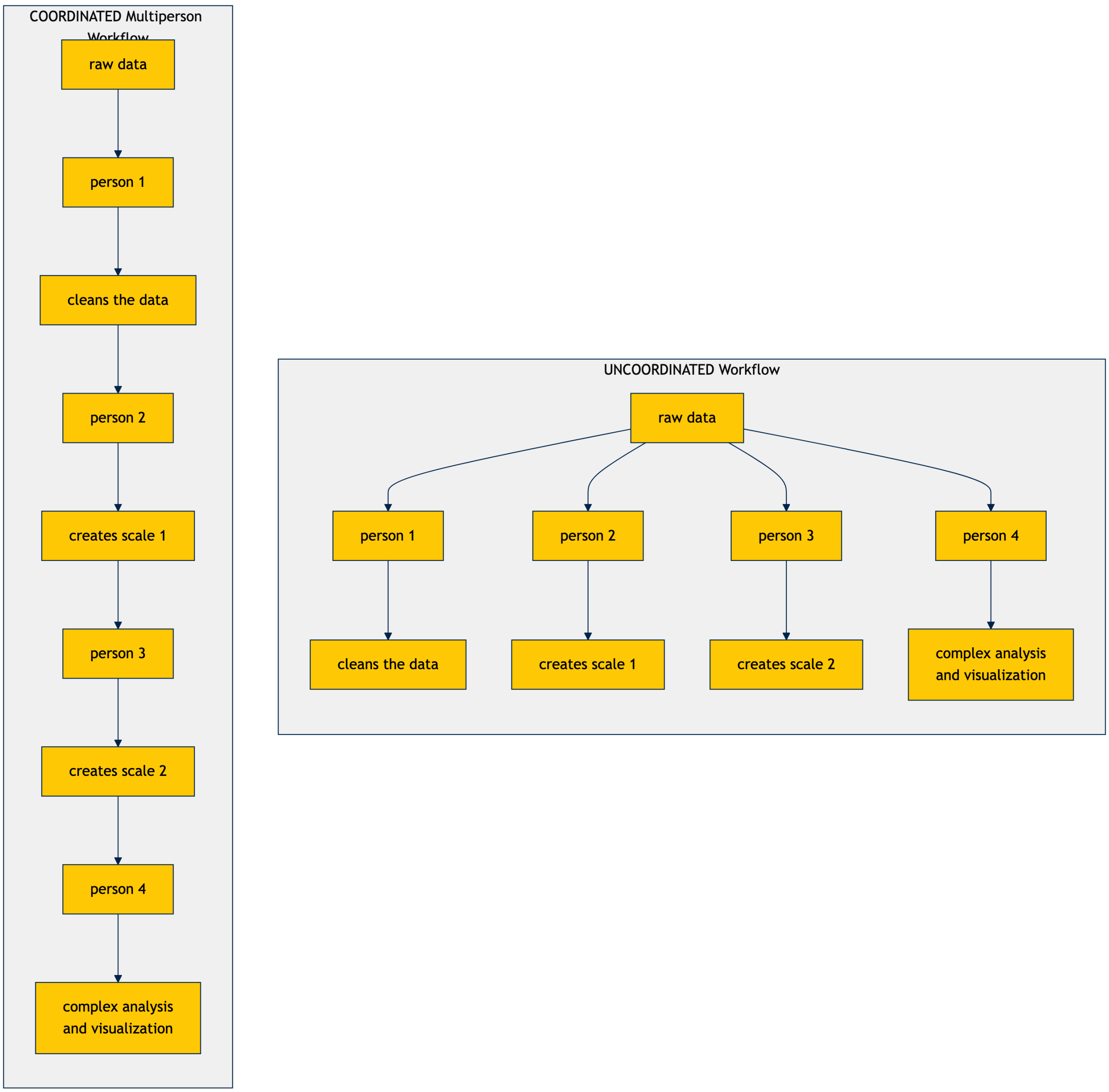

Figure 2: Complex Data Workflows