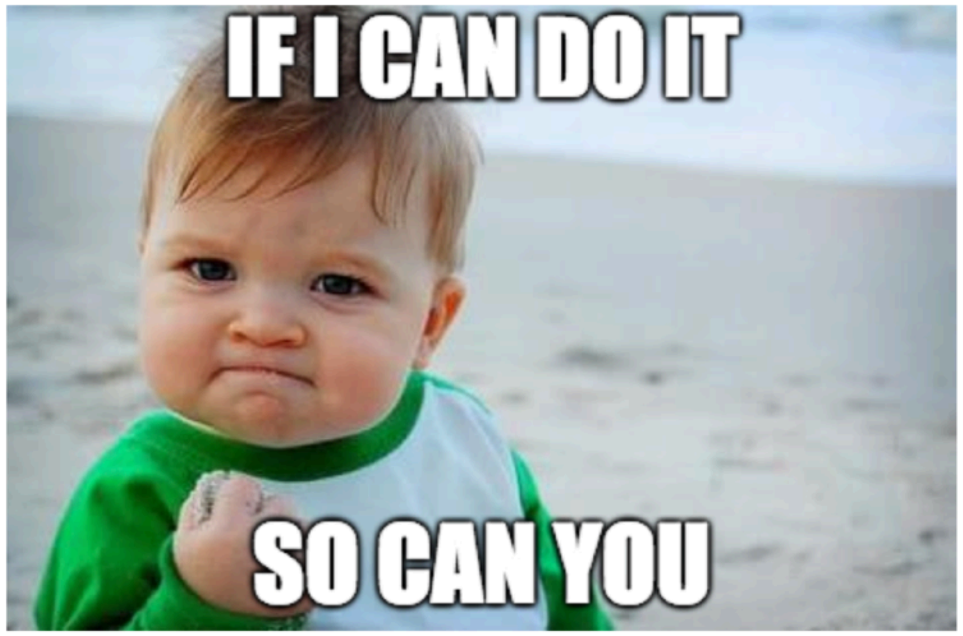
An introduction to living documents and reproducible manuscripts

Alexandra Paxton
Department of Psychological Sciences
University of Connecticut

github.com/a-paxton/living-documents

EDULANG 2022 • Storrs, CT

A bit about me



I started with
a liberal arts B.A.
(psychology and English)

(hint: not a quant)

got accepted into
Clinical Psych PhD

(still not a quant)

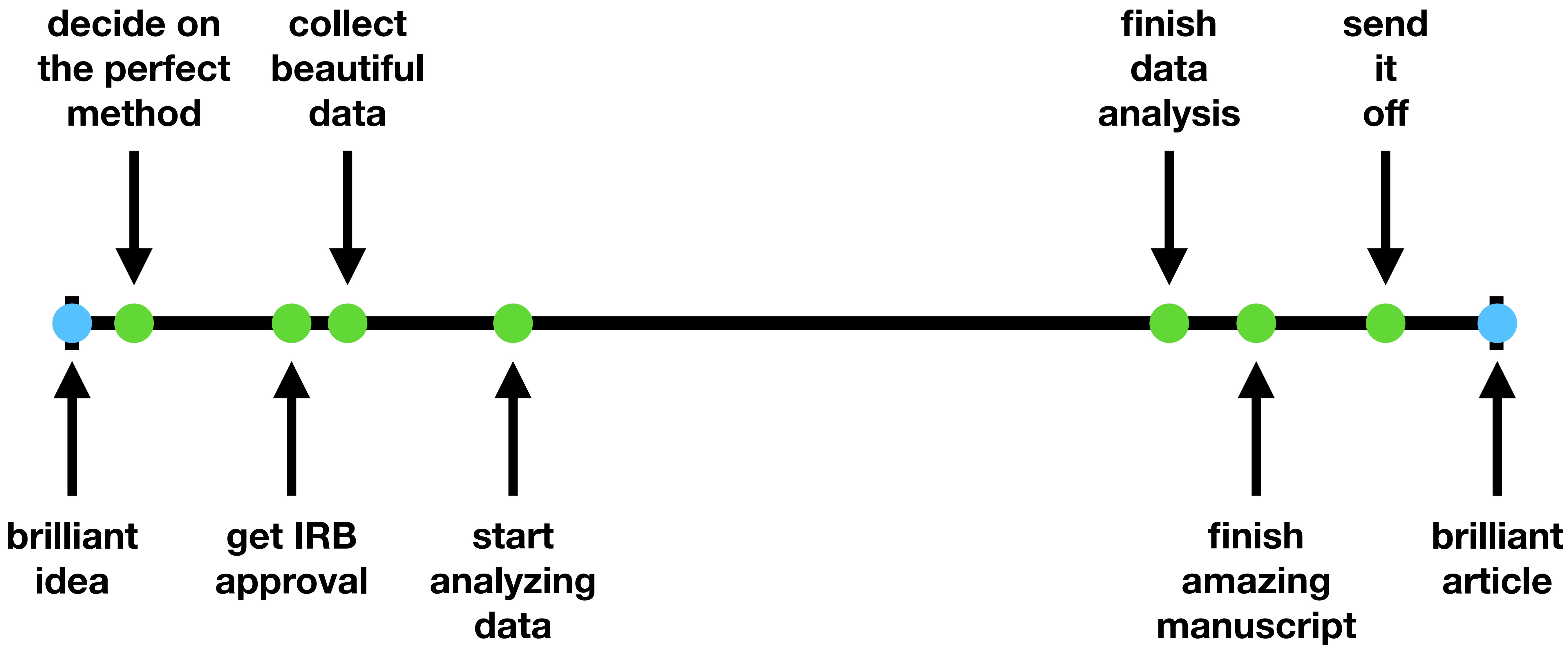
fell in love with research
as a grad student

(started my computational/quantitative journey)

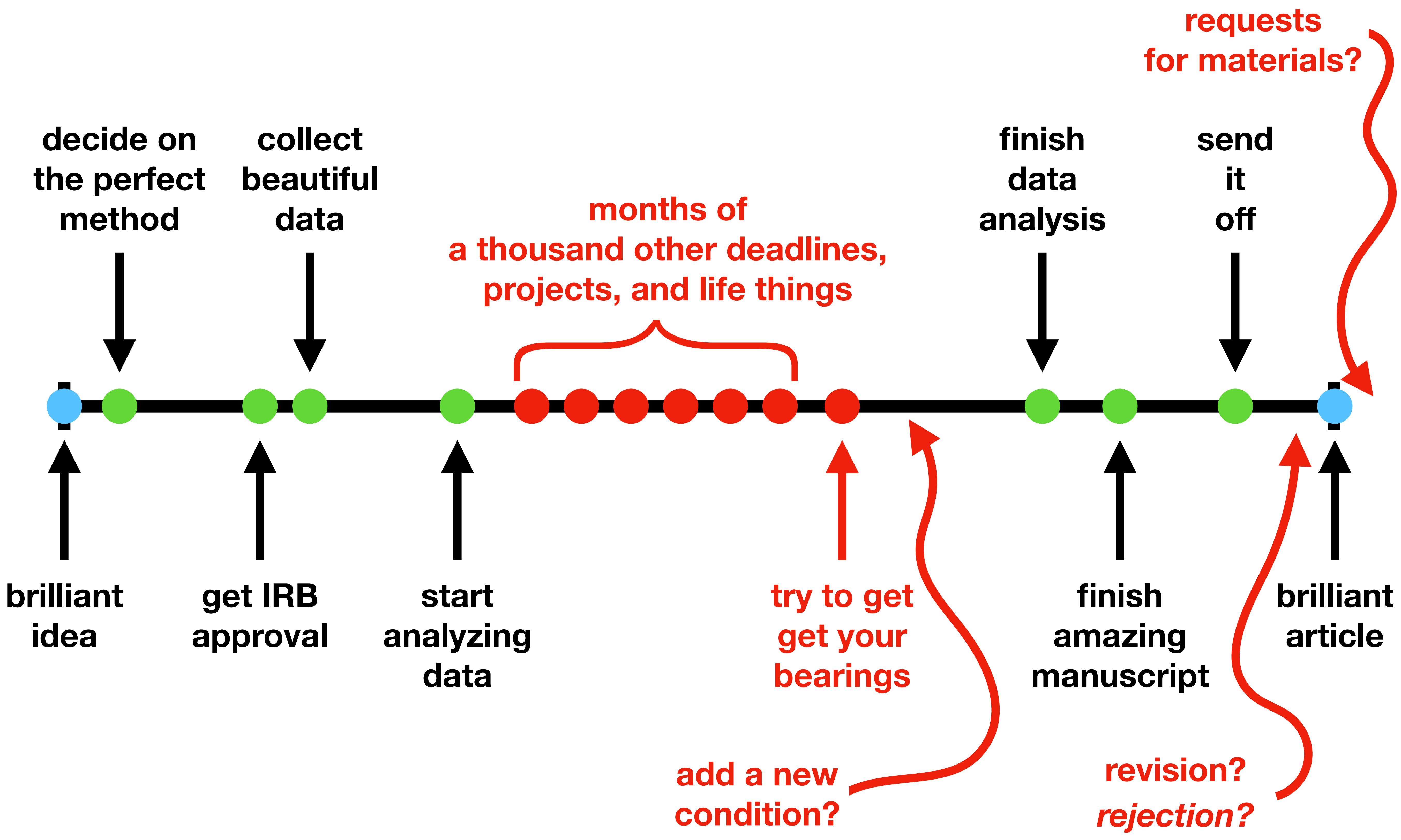
… and found myself
as a cognitive scientist
and data scientist
working to expand access
to computational methods!

(and still learning every single day!)

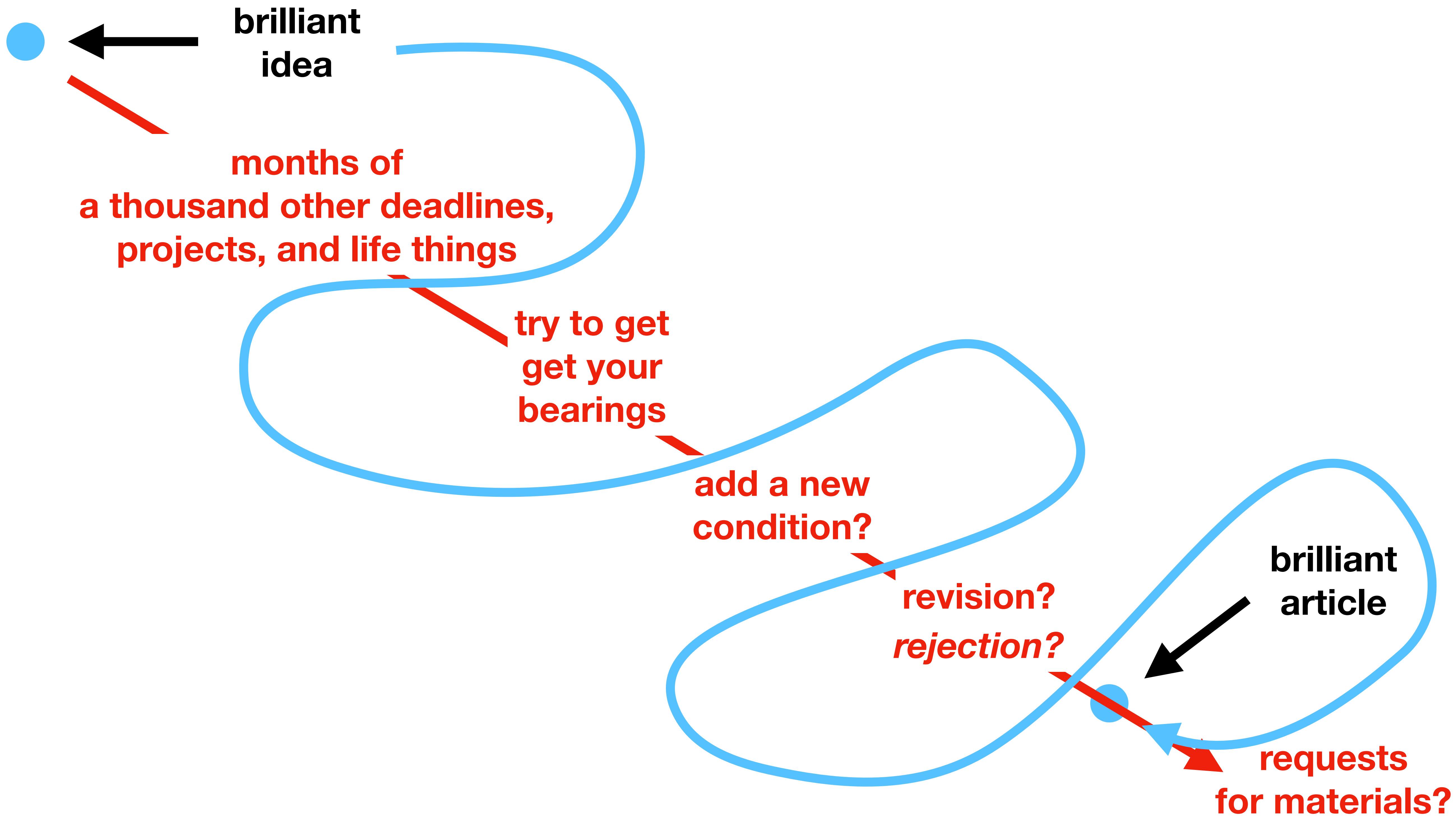
The ideal research timeline



A realistic research timeline



Research pain points



Tips for minimizing research pain points

scientific programming

use a programming language like R or Python to do your data cleaning, preparation, and analysis

living documents

keep track of code, data, and analysis choices as you build your research pipeline

reproducible manuscripts

create your manuscript with your statistics, tables, and figures all in a single platform

Why should I use…

scientific programming

creates a reproducible trace of your **data** processes

easily applies existing pipeline to **new data**

free and open-source alternatives to stats software

living documents

reproducible manuscripts

Why should I use…

scientific programming

creates a reproducible trace of your **data** processes

easily applies existing pipeline to **new data**

free and open-source alternatives to stats software

living documents

leave yourself a trace of your **research** process

document your entire research pipeline

serve as a foundation for **eventual paper writing**

reproducible manuscripts

Why should I use…

scientific programming

creates a reproducible trace of your **data** processes

easily applies existing pipeline to **new data**

free and open-source alternatives to stats software

living documents

leave yourself a trace of your **research** process

document your entire research pipeline

serve as a foundation for **eventual paper writing**

reproducible manuscripts

leave **others** a trace of your research process

share your entire research pipeline

auto-magically populate your stats, tables, and figures

Why should I use…

scientific programming

living documents

reproducible manuscripts

synergistic life savers

that will save you

time *energy* *stress*

Why should I use…

scientific programming

living documents

reproducible manuscripts

amazing scientific tools

that will improve your

reproducibility *impact* *transparency*

Why should I use…

scientific programming

living documents

reproducible manuscripts

use them for
yourself

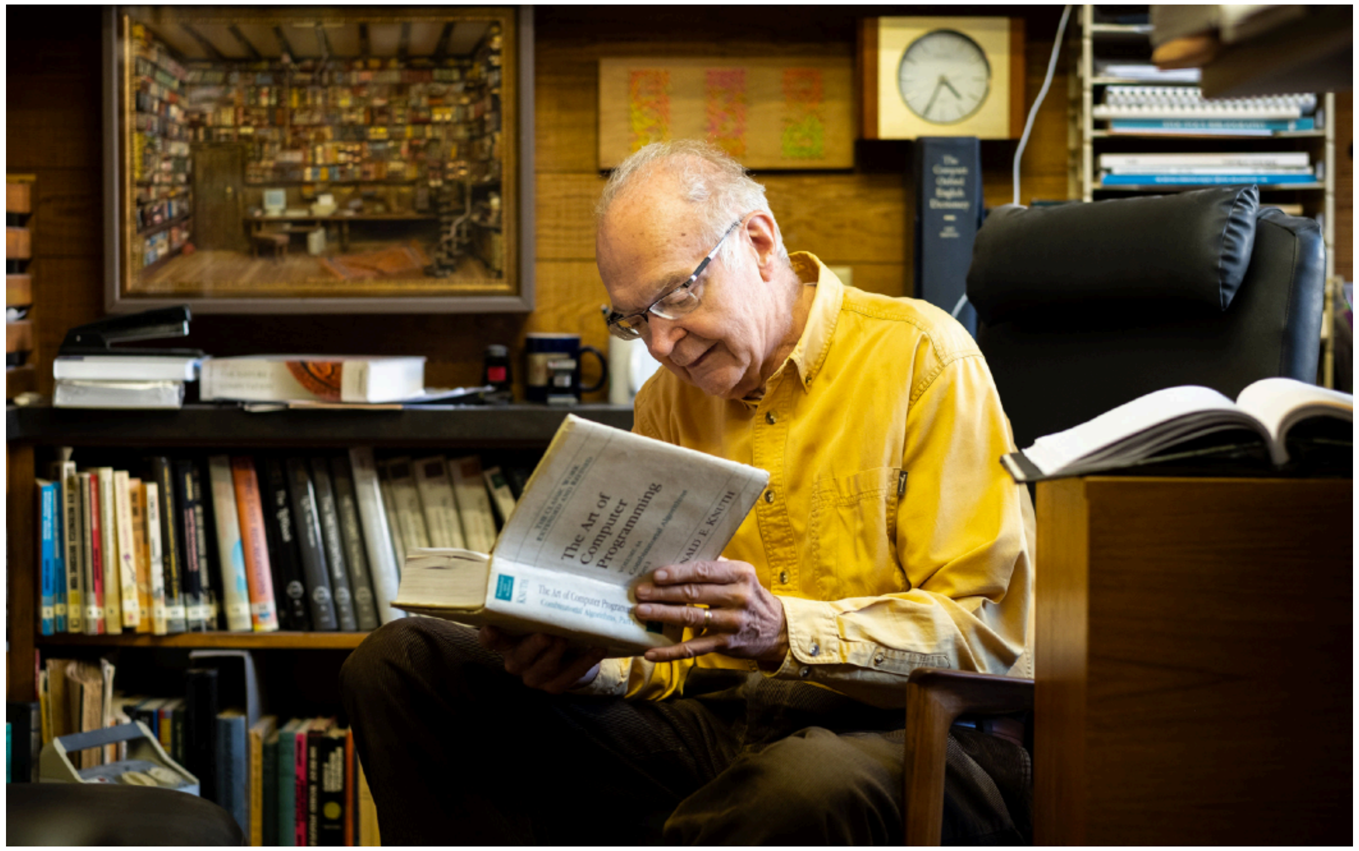
and

use them for
your science

The liberating principles of *literate programming*

“ I believe that the time is ripe for significantly better documentation of programs, and that we can best achieve this by **considering programs to be works of literature.** Hence, my title: "Literate Programming.”

Let us change our traditional attitude to the construction of programs: Instead of imagining that our main task is to instruct a computer what to do, **let us concentrate rather on explaining to human beings what we want a computer to do.** ”



Donald Knuth
creator of literate programming

Literate programming in practice

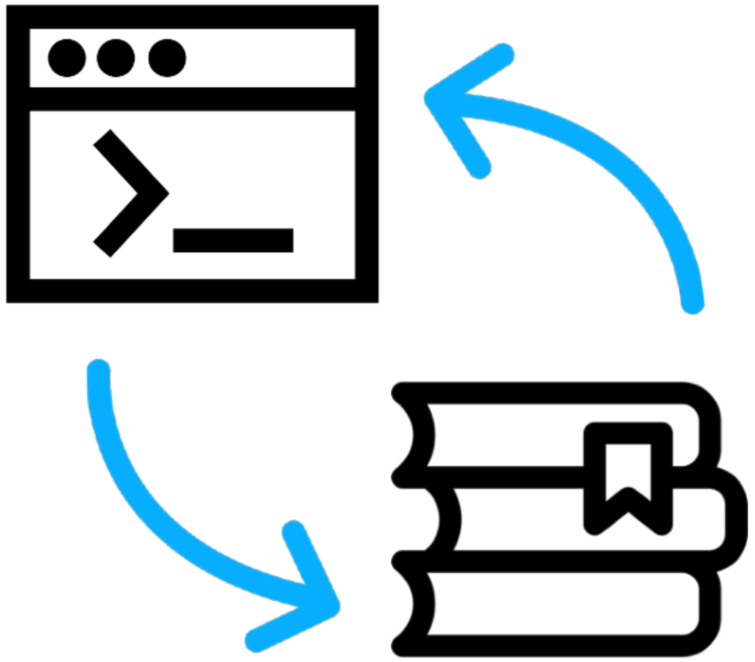
(for living documents and beyond)

comment liberally

explain the why *of the code—the functions will explain the* how *on their own*

choose names thoughtfully

in 6 months, you don't want to have to ask "what's in `df`*?" or "how is* `x` *different from* `x1`*?"*



code in "paragraphs"

break your code into more goal-oriented clusters, explaining the "why" before *presenting the code*

simple > complex

keep your code as straightforward as possible (while still being readable)

use short lines

where possible, break up your code across multiple lines (e.g., at arguments)

structure is your friend

well-named original functions and global variables can reduce transcription errors and improve readability

A reminder to cultivate a *growth-focused mindset*

free yourself of the pernicious expectations of perfection

*messy code? unclear comments?
we've all been there!*

aim for consistent improvement over a career-long journey

try adding just one new step *toward
open science with each new project*

embrace community

*take comfort that you're not alone in
discovering new practices—and
embrace new purpose in helping others*

What can I try next?

share everything
with a citable DOI via the
Open Science Framework

collaboratively develop
(and then easily share)
code on **GitHub**

license your code so that
others can use, adapt, and
build on your science

let folks run your
reproducible manuscript
in the cloud with **binder**

consider a journal that
publishes reproducible
manuscripts

GitHub repository

http://www.github.com/a-paxton/living-documents

first, we'll take a look at some **examples**…

…and then we'll **try out some** for ourselves!

(no installation required)