as clear as mud

on “good" code, visualizations, and slides

paolo adajar (they/them)
paoloadajar@mit.edu

march 11, 2025

blueprint lunch+learn
v1.1 (latest here)

Has this ever happened to you?

- You open up a code file from a project from 4 years ago. It is complete gibberish. Might as well been written in hieroglyphics.
- You're sitting in a presentation. You zone out a little bit, and look back up and have no idea what's happening. You're lost for the next 45 minutes.
- There's a missing parenthesis in your code somewhere. But just *where*...
- Your PI wants to see results with a different sample. Time to open up 20 different files and change the sample restrictions.
- You see a table or graph and have no idea what it's trying to say.

goal: provide you with tools
to think about clear communication
in coding and presentations

Why should you care?

- Working with others (including past and future you) is best when you have a shared understanding
- Transparency, reproducibility, and trust all go hand-in-hand in this profession
- Communicate to others what your work *really* is about
- Making sure your analysis is correct
- **Above all: keeps the focus on the big-picture questions**

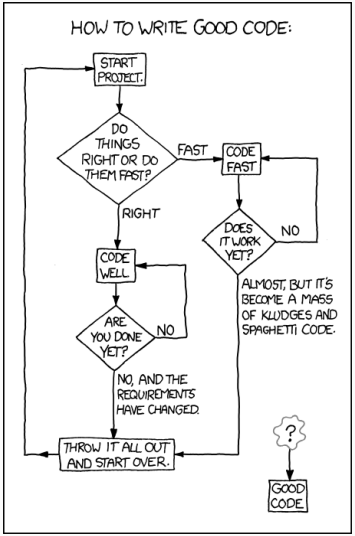
guiding philosophy:
you want people to understand.
design with that in mind.

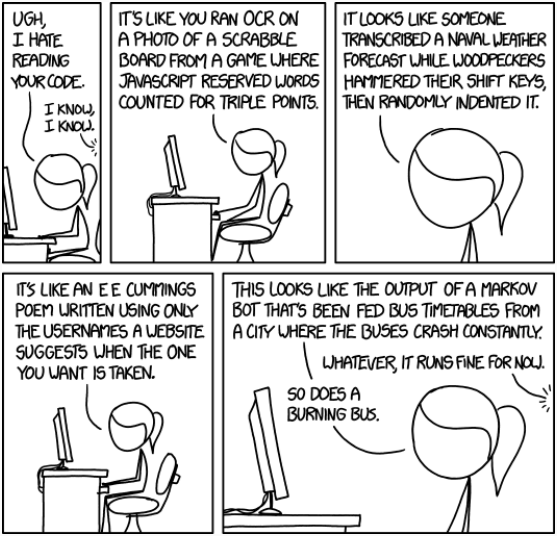
Some caveats before we begin...

- Like in most things, there are many possible “right" answers; but getting there means you need to recognize the wrong and *why* it’s wrong[1]
- There’s no way this could be comprehensive; while there will be specific tips, I could certainly talk about this for much longer
- Your audience matters; the exact implementation of design will differ massively between presentations to PIs and presentations to practitioners
- Using my own work / work of some peers; the goal is not to poke at mistakes, but to show how minor changes can make large improvements

[1] I’m also not trying to reinvent the wheel. At the end of the presentation, I link to dozens of guides on various topics.

There's always a relevant XKCD (or two)





We can take notes from people who just program for a living

The framework of 6.102 (prev. 6.031) Software Construction: create code that is

safe from bugs	correctness (in the present) and defensiveness (in the future)
easy to understand	what gets communicated to future programmers
ready for change	unfortunately, this probably won't be the last time you touch this code

Everybody starts somewhere; my first project was *not* good

```
use "D:\filepath\district_main_pol3.dta" // load the file
eststo clear // clear previous regressions
// make summary statistics for overall
eststo: estpost summarize mn_all_math mn_all_ela num_charter schools ///
totenrl100 sesall urban suburb town rural perwht perblk perhsp ///
perasn perind, detail
// make summary statistics for noncharter
eststo: estpost summarize mn_all_math mn_all_ela num_charter schools ///
totenrl100 sesall urban suburb town rural perwht perblk perhsp ///
perasn perind if num_charter == 0, detail
// make summary statistics for charter
eststo: estpost summarize mn_all_math mn_all_ela num_charter schools ///
totenrl100 sesall urban suburb town rural perwht perblk perhsp ///
perasn perind if num_charter > 0, detail
```

How I'd write the exact same code today

```
// Generate summary statistics (Table 1)
use "${cleandata}/district_policy_merged_all.dta"

assert num_charter >= 0
gen    has_charter = num_charter > 0

local covariates mn_all_math mn_all_ela                    ///
                 num_charter schools totenrl100 sesall     ///
                 urban suburb town rural                   ///
                 perwht perblk perhsp perasn perind

eststo clear
eststo: estpost summarize `covariates'                , detail
eststo: estpost summarize `covariates' if  has_charter, detail
eststo: estpost summarize `covariates' if !has_charter, detail
```

DRY: Don’t repeat yourself

- For variables, constants: ensures that there is a single, unambiguous “source of truth” for information (i.e., no “magic numbers”)
- For repeated tasks, create a function or a loop

safe from bugs	hard to notice if one variable missing in a different regression, or one iteration is different
easy to understand	fewer words in the coding file, and with appropriate names, you understand semantically what is happening
ready for change	easily change specifications, methods, without copy-paste

Much repeated code in the original version

```
use "D:\filepath\district_main_pol3.dta" // load the file
eststo clear // clear previous regressions
// make summary statistics for overall
eststo: estpost summarize mn_all_math mn_all_ela num_charter schools ///
totenrl100 sesall urban suburb town rural perwht perblk perhsp ///
perasn perind, detail
// make summary statistics for noncharter
eststo: estpost summarize mn_all_math mn_all_ela num_charter schools ///
totenrl100 sesall urban suburb town rural perwht perblk perhsp ///
perasn perind if num_charter == 0, detail
// make summary statistics for charter
eststo: estpost summarize mn_all_math mn_all_ela num_charter schools ///
totenrl100 sesall urban suburb town rural perwht perblk perhsp ///
perasn perind if num_charter > 0, detail
```

Revision defines variables to analyze, and refers to that as “truth”

```
1  // Generate summary statistics (Table 1)
2  use "${cleandata}/district_policy_merged_all.dta"
3
4  assert num_charter >= 0
5  gen    has_charter = num_charter > 0
6
7  local covariates mn_all_math mn_all_ela                    ///
8                   num_charter schools totenrl100 sesall     ///
9                   urban suburb town rural                   ///
10                  perwht perblk perhsp perasn perind
11
12 eststo clear
13 eststo: estpost summarize ‘covariates’                , detail
14 eststo: estpost summarize ‘covariates’ if  has_charter, detail
15 eststo: estpost summarize ‘covariates’ if !has_charter, detail
```

But don’t go overboard forcing things into a loop

This loop makes the code less readable

```
local cond1 if  has_charter >= 0
local cond2 if  has_charter
local cond3 if !has_charter

local conds cond1 cond2 cond3

foreach cond of local conds {
    eststo: estpost summarize ‘covariates’ if ‘‘cond’’, detail
}
```

Global filepaths help with collaboration, change

```
1  // Generate summary statistics (Table 1)
2  use "${cleandata}/district_policy_merged_all.dta"
3
4  assert num_charter >= 0
5  gen    has_charter = num_charter > 0
6
7  local covariates mn_all_math mn_all_ela                    ///
8                   num_charter schools totenrl100 sesall     ///
9                   urban suburb town rural                   ///
10                  perwht perblk perhsp perasn perind
11
12 eststo clear
13 eststo: estpost summarize `covariates'                , detail
14 eststo: estpost summarize `covariates' if  has_charter, detail
15 eststo: estpost summarize `covariates' if !has_charter, detail
```

Use well-placed assertions to help you “fail fast"

Useful Assertion

```
1 // Generate summary statistics (Table 1)
2 use "${cleandata}/district_policy_merged_all.dta"
3
4 assert num_charter >= 0
5 gen    has_charter = num_charter > 0
```

Unneeded assertion

```
1 // Generate summary statistics (Table 1)
2 use "${cleandata}/district_policy_merged_all.dta"
3
4 assert num_charter >= 0
5 gen    has_charter = num_charter > 0
6 assert has_charter == 0 | has_charter == 1
```

Use meaningful variable names, labels, to give code semantic meaning

- My original code used `charter` instead of `num_charter`, a much less helpful name
- Compare `tmp = 86400` and `secondsPerDay = 86400`
- Use your language's norms of camelCase, CapitalCase, lowercase_underscores, UPPERCASE_WITH_UNDERSCORES (often globals[2])

[2] Also–don't make something a global when it doesn't need to be!

Use whitespace to help the reader

Bad whitespace leads to gibberish (even if code's correct)

```
local covariates mn_all_math mn_all_ela num_charter schools ///
totenrl100 sesall urban suburb town rural perwht perblk perhsp ///
perasn perind
```

Vertical alignment improves comprehension

```
local covariates mn_all_math mn_all_ela                    ///
                 num_charter schools totenrl100 sesall     ///
                 urban suburb town rural                   ///
                 perwht perblk perhsp perasn perind
```

Related: limit lines to 80 characters!

Comments are great, but you shouldn't rely on them to convey meaning

```
use "D:\filepath\district_main_pol3.dta" // load the file
eststo clear // clear previous regressions
// make summary statistics for overall
eststo: estpost summarize mn_all_math mn_all_ela num_charter schools ///
totenrl100 sesall urban suburb town rural perwht perblk perhsp ///
perasn perind, detail
// make summary statistics for noncharter
eststo: estpost summarize mn_all_math mn_all_ela num_charter schools ///
totenrl100 sesall urban suburb town rural perwht perblk perhsp ///
perasn perind if num_charter == 0, detail
// make summary statistics for charter
eststo: estpost summarize mn_all_math mn_all_ela num_charter schools ///
totenrl100 sesall urban suburb town rural perwht perblk perhsp ///
perasn perind if num_charter > 0, detail
```

The best code is self-commenting

```
1  // Generate summary statistics (Table 1)
2  use "${cleandata}/district_policy_merged_all.dta"
3
4  assert num_charter >= 0
5  gen    has_charter = num_charter > 0
6
7  local covariates mn_all_math mn_all_ela                    ///
8                   num_charter schools totenrl100 sesall     ///
9                   urban suburb town rural                   ///
10                  perwht perblk perhsp perasn perind
11
12 eststo clear
13 eststo: estpost summarize `covariates'                , detail
14 eststo: estpost summarize `covariates' if  has_charter, detail
15 eststo: estpost summarize `covariates' if !has_charter, detail
```

Tips for self-commenting code

- Use larger comments at the beginning of files, sections, functions, to describe their purpose (and if relevant, inputs and outputs)
- Semantic names for variables, commands, and whitespace can make your code read like English
- Each file should broadly be serving one purpose (which is why in Blueprint, we divide things between `clean`, `build`, and `analysis`)

It can be hard to do this all of the time

- Oftentimes, especially when coding fast or creating a proof-of-concept, your code will not be very clean on the first run
- The goal is not to be perfect all of the time; the goal is to start with principles in mind, and change things as you go along
- *Refactoring* code is extraordinarily helpful, especially for long-term projects
- For the smaller things, using a *linter* can help make sure your code follows a standard style[3]

[3] To be quite honest, for every language, there's a million different guides on how the three principles should be applied. What's most important is that you keep these principles in mind and be consistent within your work.

Other tiny coding tips

- Always use forward slashes (/) in filepaths, not backslashes (\); backslashes work only on Windows and are escape characters in many languages
- Use version control (e.g., Git) to track changes and collaborate with others
- For replicability, manage the versions of packages, libraries used in your code

safe from bugs.

easy to understand.

ready for change.

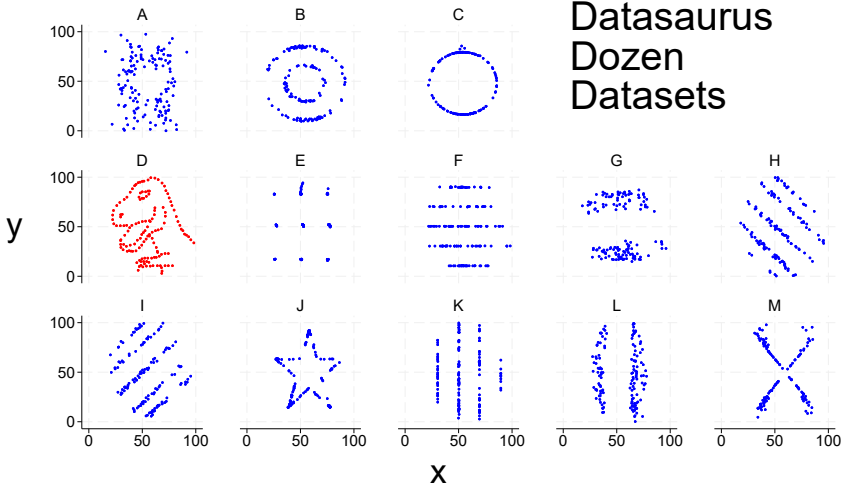
Visual representations are a blessing and a curse

- In a presentation, most results will be framed as a graphic or a table
- Default settings are a one-size-fits-all solution, but won't be perfect for most use cases
- How to think about making these as interpretable as possible?

There are definitely bad ways to present data

Dataset	$\mathbb{E}[X]$	$SD[X]$	$\mathbb{E}[Y]$	$SD[Y]$	Regression	r^2	N
A	54.27	16.77	47.83	26.94	y = −0.10x + 53.43	0.004	142
B	54.27	16.77	47.83	26.94	y = −0.11x + 53.81	0.005	142
C	54.27	16.76	47.84	26.93	y = −0.11x + 53.80	0.005	142
D	54.26	16.77	47.83	26.94	y = −0.10x + 53.45	0.004	142
E	54.26	16.77	47.84	26.93	y = −0.10x + 53.10	0.004	142
F	54.26	16.77	47.83	26.94	y = −0.10x + 53.21	0.004	142
G	54.27	16.77	47.84	26.94	y = −0.11x + 53.81	0.005	142
H	54.27	16.77	47.84	26.94	y = −0.11x + 53.85	0.005	142
I	54.27	16.77	47.83	26.94	y = −0.11x + 53.81	0.005	142
J	54.27	16.77	47.84	26.93	y = −0.10x + 53.33	0.004	142
K	54.27	16.77	47.84	26.94	y = −0.11x + 53.89	0.005	142
L	54.27	16.77	47.83	26.94	y = −0.11x + 53.63	0.004	142
M	54.26	16.77	47.84	26.93	y = −0.11x + 53.55	0.004	142

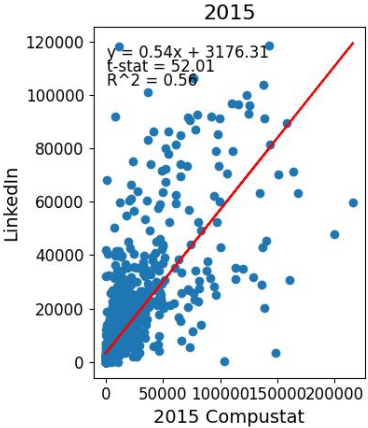
And there are ways that are far more memorable

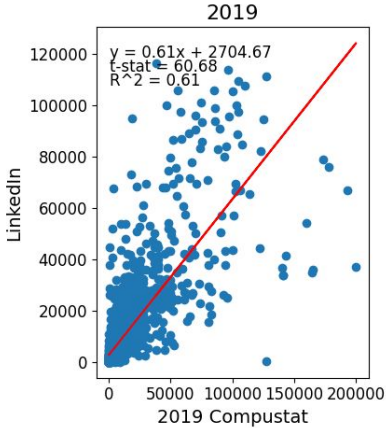


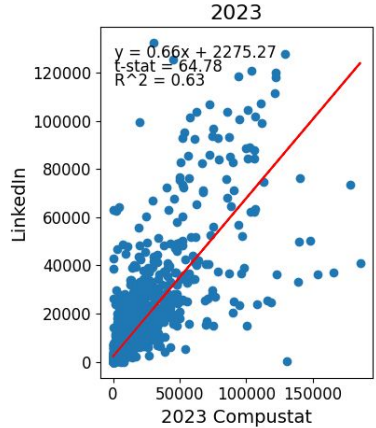
Five principles of good graphs, from Schwabish (2021)

1. Show the data
2. Reduce the clutter
3. Integrate the graphics and text
4. Avoid the spaghetti chart
5. Start with gray

Revelio correlates with Compustat employment

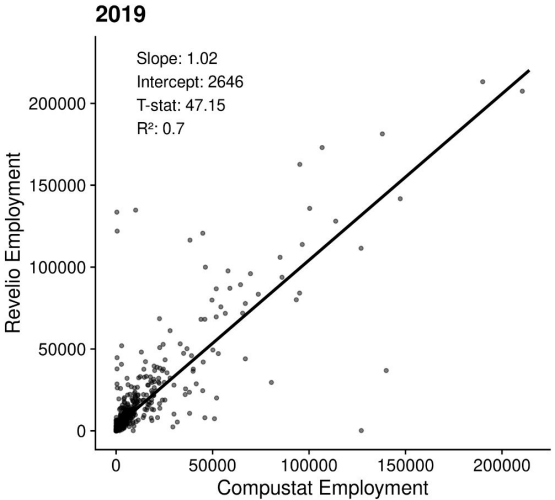


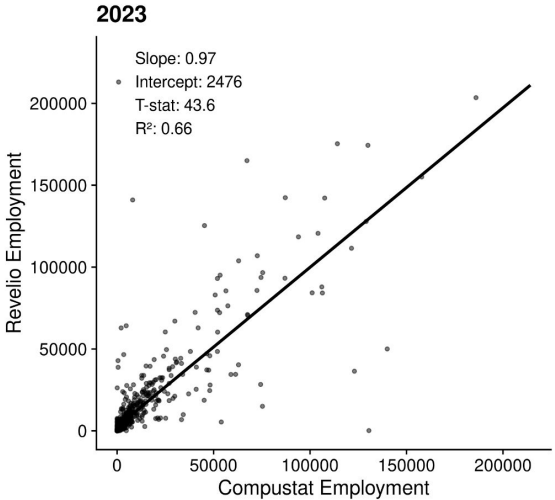




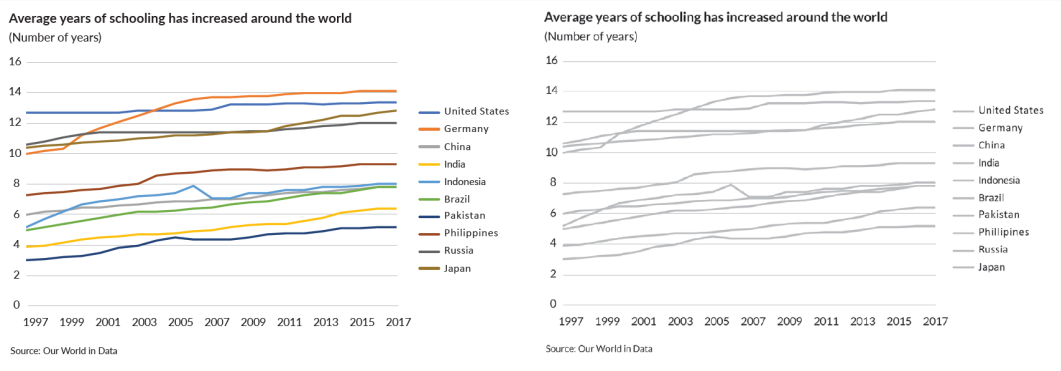
In Logs

Validation: Revelio vs. Compustat employment



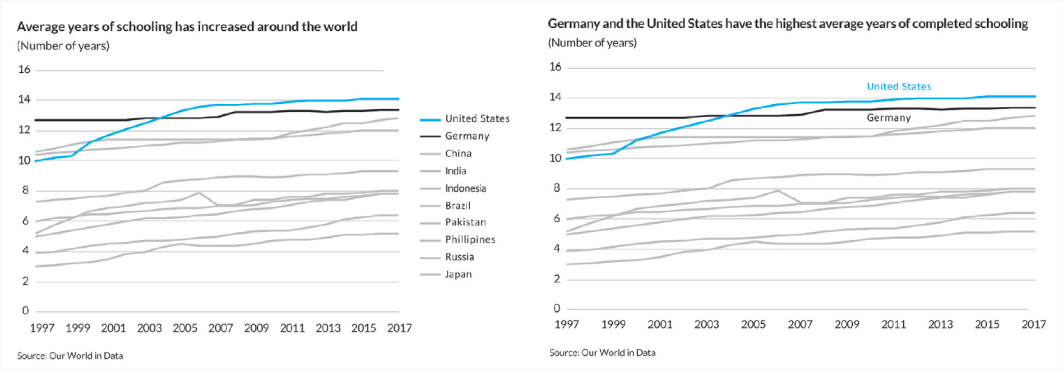


When graphs are too busy, you lose the story



Source: Schwabish (2021)

Purposeful colors, in-graph labels, and informative titles create clarity



Source: Schwabish (2021)

Regression results can be shown in many different ways (bad table)

	Model 1	Model 2	Model 3
r_age	0.0509***	0.0119***	0.0207***
	(0.0062)	(0.0044)	(0.0026)
gndr	0.0442***	0.0616***	0.0630***
	(0.0057)	(0.0037)	(0.0043)
_educ	0.0027***	0.0052***	0.0157***
	(0.0087)	(0.0050)	(0.0072)
hrswkd	0.0397***	0.0075***	0.0211***
	(0.0053)	(0.0025)	(0.0029)
expr	0.0003***	0.0043***	0.0030***
	(0.0051)	(0.0026)	(0.0024)
marstat	0.0191***	0.0066***	0.0069***
	(0.0053)	(0.0025)	(0.0027)

* p < 0.05, ** p < 0.01, *** p < 0.001

Source: Schwabish (2021)

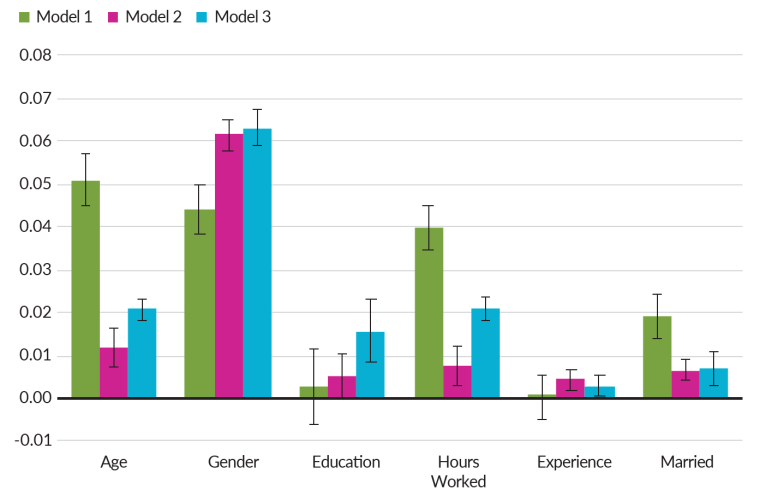
Regression results can be shown in many different ways (better table)

	Model 1	Model 2	Model 3
Age	0.0509***	0.0119***	0.0207***
	(0.0062)	(0.0044)	(0.0026)
Gender	0.0442***	0.0616***	-0.0630***
	(0.0057)	(0.0037)	(0.0043)
Education	0.0027	0.0052	0.0157**
	(0.0087)	(0.0050)	(0.0072)
Hours Worked	0.0397***	0.0075*	0.0211***
	(0.0053)	(0.0044)	(0.0029)
Experience	0.0003	0.0043*	0.0030
	(0.0051)	(0.0026)	(0.0024)
Married	0.0191***	0.0066***	0.0069*
	(0.0053)	(0.0025)	(0.0041)

* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

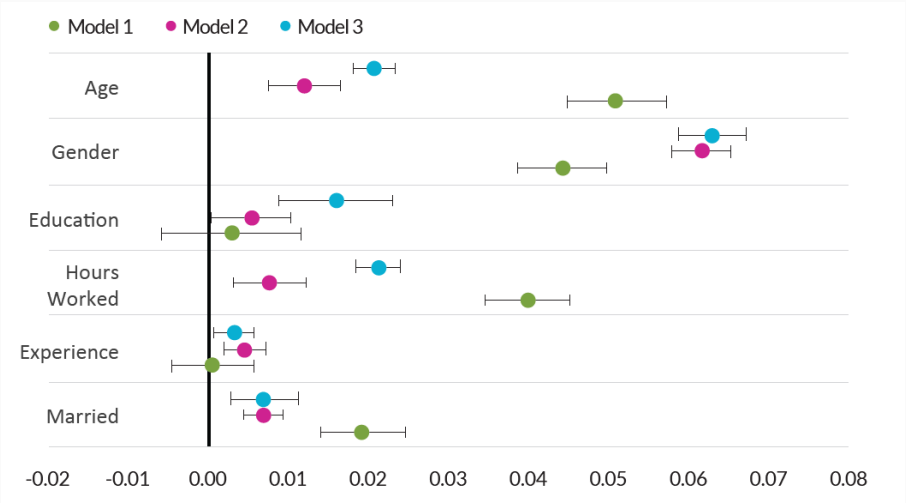
Source: Schwabish (2021)

Regression results can be shown in many different ways (bar chart)



Source: Schwabish (2021)

Regression results can be shown in many different ways (dot chart)



Source: Schwabish (2021)

Good ideas aren’t useful if they can’t be communicated well

- After putting hundreds of hours into a project, the last thing you want is for your ideas to be misunderstood
- Slides often created as an afterthought after analysis or writing (how many times have you seen a presentation with a table copy-pasted from the paper?)
- What story do you want to tell? Is that story easy to find?

The High Cost of Managing Data

These regulations increase the firms' **cost of data** and **distort** the input choices

- Generates a wedge between the marginal product of data and its price
- Leads to misallocation if the compliance cost varies across firms (Hsieh and Klenow, 2009)

- How privacy laws distort firms input choices, except for digital goods, is not well understood
- Requires a framework to analyze how firms use and process data

This paper:

1. How do firms combine data and computation in production?
2. What is the cost of GDPR for firms and how do they change their data/computation inputs?

Attention is precious; be judicious in where you ask for it

Design for the person who zoned out of your presentation and wants to tune back in

- The more you emphasize on a slide, the less useful it is
- Use a consistent emphasis style
- Slide titles should be sentences that summarize the slide (and so each slide should make one point)
- At *most*, use two levels of bullets (and many times, they're optional)

These regulations increase firms' cost of data and distort input choices

- Privacy laws like GDPR generate a wedge between marginal product of data and its price
- Leads to misallocation if compliance cost varies across firms (Hsieh and Klenow, 2009)
- Distortion not well understood; requires framework of how firms use and process data

This paper:

1. How do firms combine data and computation in production?
2. What is the cost of GDPR for firms?
3. How do firms change their data/computation inputs?

Another framing: slides should guide viewers to the desired conclusion

- Easy to get lost in tables, graphs, dense paragraphs of text
- Design your exhibits so that they tell the story without you[4]
- Any fancy graphics should *help* the reader, not confuse
- Don't overfill slides; details are good for slides that'll be referenced, not for presenting

[4] An ethos much like self-commenting code.

Teacher "nominations" are more likely from scheduled castes, tribes

	All	Schools Selected	Exam Select	Teacher Top	Teacher Nom	Rand
Test scores	66.18	63.93	89.66	83.74	71.96	76.81
	(16.08)	(15.01)	(6.299)	(12.27)	(11.82)	(8.504)
English medium	0.281	0.348	0.368	0.350	0.363	0.344
	(0.450)	(0.476)	(0.482)	(0.477)	(0.481)	(0.476)
Scheduled Caste	0.344	0.322	0.255	0.246	0.282	0.302
	(0.475)	(0.467)	(0.436)	(0.431)	(0.450)	(0.460)
Scheduled Tribe	0.0383	0.0272	0.0137	0.0139	0.0161	0.0153
	(0.192)	(0.163)	(0.116)	(0.117)	(0.126)	(0.123)
Female	0.513	0.537	0.665	0.642	0.535	0.645
	(0.500)	(0.499)	(0.472)	(0.480)	(0.499)	(0.479)
Mom Ed.	3.101	3.156	2.788	2.794	2.865	3.096
	(4.016)	(4.006)	(3.643)	(3.564)	(3.740)	(3.910)
Dad Ed.	2.813	2.880	2.571	2.612	2.558	2.824
	(3.888)	(3.894)	(3.572)	(3.548)	(3.616)	(3.816)
N	102316	65808	6212	3384	2182	262

Table reports means of non-missing observations in subgroup, SD in parentheses. Source: TN SED

Screen 1: Teacher “nominations” are more diverse than their “top”

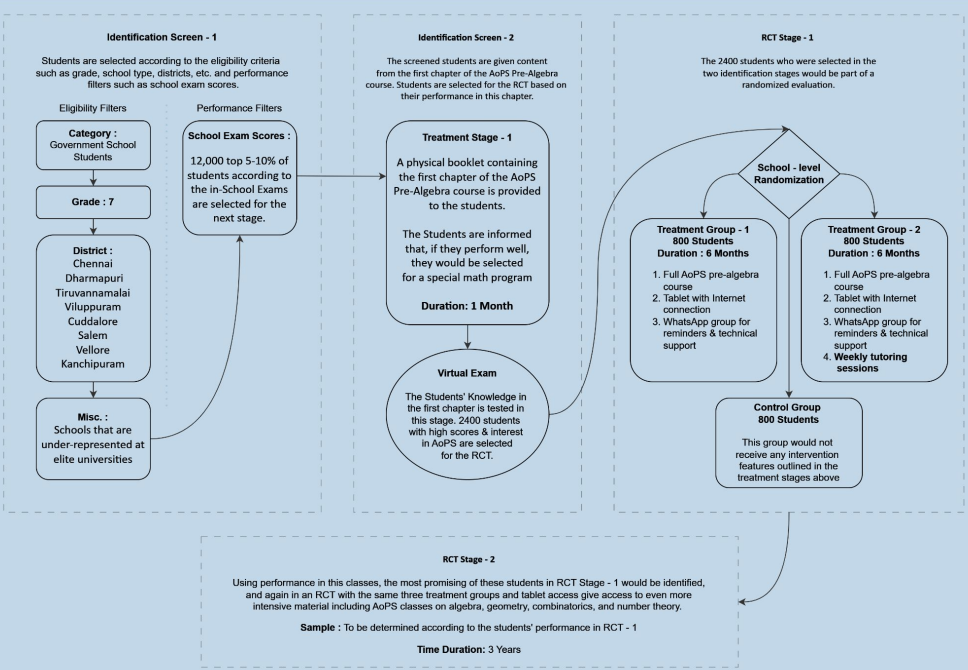
	All	Schools Selected	Exam Select	Teacher Top	Teacher Nom	Rand
Test scores	66.18	63.93	89.66	83.74	71.96	76.81
English medium	0.281	0.348	0.368	0.350	0.363	0.344
Scheduled Caste	0.344	0.322	0.255	0.246	0.282	0.302
Scheduled Tribe	0.0383	0.0272	0.0137	0.0139	0.0161	0.0153
Female	0.513	0.537	0.665	0.642	0.535	0.645
Mom Ed.	3.101	3.156	2.788	2.794	2.865	3.096
Dad Ed.	2.813	2.880	2.571	2.612	2.558	2.824
N	102316	65808	6212	3384	2182	262

Table reports means of non-missing observations in subgroup, SD in parentheses. Source: TN SED data.

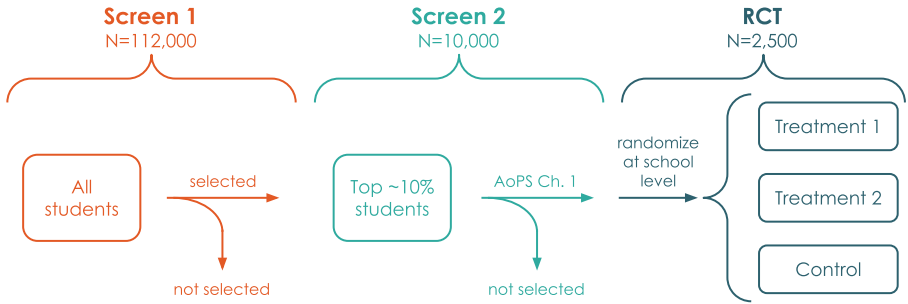
Another framing: slides should guide viewers to the desired conclusion

- Easy to get lost in tables, graphs, dense paragraphs of text
- Design your exhibits so that they tell the story without you[2]
- Any fancy graphics should *help* the reader, not confuse
- Don't overfill slides; details are good for slides that'll be referenced, not for presenting

[2] An ethos much like self-commenting code.


Identification Screen - 1
Students are selected according to the eligibility criteria such as grade, school type, districts, etc. and performance filters such as school exam scores.
Eligibility Filters
Performance Filters
Category : Government School Students
Grade : 7
District : Chennai Dharmapuri Tiruvannamalai Viluppuram Cuddalore Salem Vellore Kanchipuram
Misc. : Schools that are under-represented at elite universities
School Exam Scores :
12,000 top 5-10% of students according to the in-School Exams are selected for the next stage.
Identification Screen - 2
The screened students are given content from the first chapter of the AoPS Pre-Algebra course. Students are selected for the RCT based on their performance in this chapter.
Treatment Stage - 1
A physical booklet containing the first chapter of the AoPS Pre-Algebra course is provided to the students.
The Students are informed that, if they perform well, they would be selected for a special math program
Duration: 1 Month
Virtual Exam
The Students' Knowledge in the first chapter is tested in this stage. 2400 students with high scores & interest in AoPS are selected for the RCT.
RCT Stage - 1
The 2400 students who were selected in the two identification stages would be part of a randomized evaluation.
School - level Randomization
Treatment Group - 1
800 Students
Duration : 6 Months
1. Full AoPS pre-algebra course
2. Tablet with Internet connection
3. WhatsApp group for reminders & technical support
Treatment Group - 2
800 Students
Duration : 6 Months
1. Full AoPS pre-algebra course
2. Tablet with Internet connection
3. WhatsApp group for reminders & technical support
4. Weekly tutoring sessions
Control Group
800 Students
This group would not receive any intervention features outlined in the treatment stages above
RCT Stage - 2
Using performance in this classes, the most promising of these students in RCT Stage - 1 would be identified, and again in an RCT with the same three treatment groups and tablet access give access to even more intensive material including AoPS classes on algebra, geometry, combinatorics, and number theory.
Sample : To be determined according to the students' performance in RCT - 1
Time Duration: 3 Years

Overview



- All 7th grade students in selected districts
- Use **school test scores, teachers,** for selection

- Offer students physical copies of **Chapter 1 of AoPS Pre-algebra**
- Screen using **admin scores, virtual exam, engagement and "grit"**

T1 AoPS pre-algebra, logistics support, and tablet
T2 Tutor + T1
Control No AoPS
\+ cross-randomized treatments…

Another framing: slides should guide viewers to the desired conclusion

- Easy to get lost in tables, graphs, dense paragraphs of text
- Design your exhibits so that they tell the story without you[2]
- Any fancy graphics should *help* the reader, not confuse
- Don't overfill slides; details are good for slides that'll be referenced, not for presenting

[2] An ethos much like self-commenting code.

Measuring remote work using job postings

- Used GPT-4o mini to classify 100 job postings per month from 2020 to 2023 per firm as either in person or remote/hybrid a la (Hansen et al 2024)
  - Today: top 50 and bottom 50 companies by change in offshore workers for expediency
  - Share of remote jobs over this period captures duration x intensity of remote work

Nvidia, Technology lead, May 2024, San Jose: “We are looking for a Machine Learning Engineer/AI Solutions Architect with experience in AI data pipelines and model development.…We make extensive use of conferencing tools, but occasional travel is required for local on-site visit to customers and data science conferences. **We are open to remote work location**…”

Nvidia, Software Engineer, Oct 2024, Santa Clara: “We are looking for a Principal Software Engineer with experience in building highly scalable and robust enterprise software to join us…If you're creative and passionate about developing services to manage a cluster of GPUs/CPUs we want to hear from you! **#LI-Hybrid** The base salary range is 272,000 USD - 419,750 USD…”

Measuring remote work using job postings

- Begin with firms that have Revelio office job postings and are in Compustat (N = 400)
  - Moving forward, will use Lightcast for better job postings coverage
  - Will also look at Scoop Flex Index (used in Gupta, Mittal, Van Nieuwerburgh 2025)
- Used GPT-4o mini to classify 100 US job postings per month per firm from January 2019 to November 2024 as either in person or remote/hybrid, à la Hansen et al 2024
  - Share of remote jobs over this period captures duration × intensity of remote work
  - On a holdout set of 100 postings GPT achieves 95% accuracy

Nvidia, Technology Lead, May 2024, San Jose: "We are looking for a Machine Learning Engineer/AI Solutions Architect with experience in AI data pipelines and model development.…We make extensive use of conferencing tools, but occasional travel is required for local on-site visit to customers and data science conferences. **We are open to remote work location**…"

Nvidia, Software Engineer, Oct 2024, Santa Clara: "We are looking for a Principal Software Engineer with experience in building highly scalable and robust enterprise software to join us…If you're creative and passionate about developing services to manage a cluster of GPUs/CPUs we want to hear from you! **#LI-Hybrid** The base salary range is 272,000 USD - 419,750 USD…"

Small details add a lot to “polish" (1)

Dashes are meaningful, so use the right one

hyphen	-	compound words	follow-up, mother-in-law
en-dash	–	ranges of numbers	Jan. 1–15, Exhibits A–E
		two words together that either connect or contrast	Dodd–Frank, urban–rural divide
em-dash	—	sentence break when commas are too weak, but parentheses, colons, or semicolons are too strong	I think that there are too many rules—more than any person should be expected to remember.
minus	−	subtraction	5−2=3

Small details add a lot to “polish" (2)

- Either every line should end in a period or none should (throughout)
- Use 16:9 aspect ratio (in LaTeX, the default is 4:3)
- Notation should be consistent (COVID or Covid-19? k, M, or billion?)

guiding philosophy:
you want people to understand.
design with that in mind.

slack or paoloadajar@mit.edu for questions and thoughts

permalink to latest version here

Resources for coding (General)

- Innovations for Poverty Action, Best Practices for Data and Code Management (link) and Cleaning Guide (link)
- Grant McDermott, EC607 Lecture Notes (link)
- Matt Gentzkow and Jesse Shapiro, Code + Data for the Social Sciences (link) and RA guide (link) (highly recommended)
- World Bank DIME Analytics Data Handbook (link)
- Arthur Turrell, Coding for Economists (link)
- Rules for open-source scientific software (link)
- MIT 6.031, Lecture 4: Code Review (link), Lecture 9: Avoiding Debugging (link), Code Review overview (link)

Stata coding resources

- Stata linter following DIME's Stata coding practices (link)
- Stata Guide, Sean Higgins (link)
- Michael Stepner, Coding Style Guide (link)

R coding resources

- Software Carpentry, Best Practices for Writing R Code (link)
- TASO, Practice Tips (link)
- R-bloggers, Best Practices (link)
- Google, R Style Guide (link) and linter (link)

Beamer and Presentation resources

- Paul Goldsmith-Pinkman, Beamer Tips for Economists (link)
- Natalia Emmanuel, A Basic Beamer Power Up (link)
- Jesse Shapiro, How to Give an Applied Micro Talk (link)
- Rachael Meager, Public Speaking for Academic Economists (link)
- Presentation Rules and Suggestions from University of Munich (link)
- Monika Piazzesi, Avoiding disasters in presentations (link)
- Tim Kehoe, Presentation Tips (link)
- Harvard Writing and Communication Center, Fundamentals of Slide Design (link)

Visualization resources

- Kieran Healy, Data Visualization: A Practical Introduction (link)
- Jonathan Schwabish, Better Data Visualizations (link)
- Arnav Bandekar, Making Economics Theory Graphs in LaTeX(link)
- Edward Tufte's books

Other resources that didn’t fit anywhere else

- Older article on reproducible economics research (link)
- Writing code in Python for Economists (link)
- Notes on typography (link)

An example of “too many words"— though the tips are real, detail comes at the expense of legibility

- Dense graphs, tables, and paragraphs can confuse the reader
  - Many graphs, tables have extraneous information. The graphics you put into a presentation should almost *never* be the same as those in your paper.
  - You don’t want to be reading from paragraphs anyways. Often, more words are used as a “crutch" by speakers.
- Design your exhibits so that they tell the story without you
  - Somewhat like self-commenting code; exhibits should speak for themselves.
  - Use highlighting, markers, to “signpost" the meaning of it (bold, different color, overlaid boxes).
- Any fancy graphics should *help* the reader, not confuse
  - Graphics should be as visually simple as possible: no outlines, allowed to have each section be one color.
  - Large blocks of color unideal for visibility on projectors.
- Don’t overfill slides; whitespace is allowed, and in fact, good
  - If you are decreasing your font size to fit the slide, there’s too much.
  - Spacing between lines and paragraphs can greatly help clarity.