Programming Camp - Lecture 1

Brian Higgins
Ciaran Rogers

This version: August 30, 2021

Welcome to the Farm!

1. Name, where you're from
2. Something uninteresting about yourself, for example
   a. What is the food everyone loves but you hate?
   b. One strange thing about your hometown?
   c. Fictional town you'd like to visit (book/tv/movie)?
   d. What's your go-to meal to cook?
   e. Something else uninteresting!

Welcome to Stanford!

What to expect from the programming camp:

- Familiarize *all* students with data processing through Stata/R/Python and MATLAB, and Stanford's servers.
- Provide a clear basis so that students can learn these programs by themselves.
- Level the playing field (so it may be slow for some).
- Make you aware of what is out there (for when you run out of RAM, or need to go 100x faster).

Welcome to Stanford!

What to expect from the programming camp:

- Familiarize *all* students with data processing through Stata/R/Python and MATLAB, and Stanford's servers.
- Provide a clear basis so that students can learn these programs by themselves.
- Level the playing field (so it may be slow for some).
- Make you aware of what is out there (for when you run out of RAM, or need to go 100x faster).

What NOT to expect from it:

- Full proficiency (sorry!) — it is impossible in our four 2-hour sessions.

Welcome to Stanford!

What to expect from the programming camp:

- Familiarize *all* students with data processing through Stata/R/Python and MATLAB, and Stanford's servers.
- Provide a clear basis so that students can learn these programs by themselves.
- Level the playing field (so it may be slow for some).
- Make you aware of what is out there (for when you run out of RAM, or need to go 100x faster).

What NOT to expect from it:

- Full proficiency (sorry!) — it is impossible in our four 2-hour sessions.

One important ground rule:

- Help each other!

Roadmap for the programming camp

1. Good coding practices and Stata (day 1)
2. MATLAB (days 2 and 3)
3. Using Stanford servers, parallel computing in MATLAB, other general advice (day 4)

Presentation outline

1. Why is coding important?
2. General Programming Advice
3. Stata

1 | Why is coding important?

General Programming Advice

Stata

Data usage has increased in Economics (Angrist et. al; 2017, AER PP)

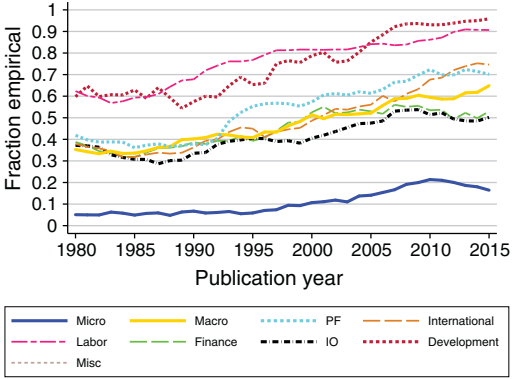


FIGURE 4. WEIGHTED FRACTION EMPIRICAL BY FIELD

Note: Five-year moving averages of the weighted fraction of publications in each field that are empirical.

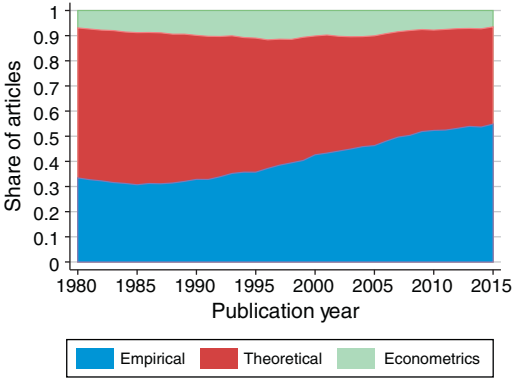


FIGURE 6. WEIGHTED PUBLICATIONS BY STYLE

Note: Five-year moving averages of weighted publication shares in each style.

Even among theorists. (!!!) Shengwu Li (2017, AER)

Obviously Strategy-Proof Mechanisms†

By Shengwu Li*

A strategy is obviously dominant if, for any deviation, at any information set where both strategies first diverge, the best outcome under the deviation is no better than the worst outcome under the dominant strategy. A mechanism is obviously strategy-proof (OSP) if it has an equilibrium in obviously dominant strategies. This has a behavioral interpretation: a strategy is obviously dominant if and only if a cognitively limited agent can recognize it as weakly dominant. It also has a classical interpretation: a choice rule is OSP-implementable if and only if it can be carried out by a social planner under a particular regime of partial commitment. (*JEL* D11, D44, D82)

Even among theorists. (!!!) Shengwu Li (2017, AER)

IV. Laboratory Experiment

Are obviously strategy-proof mechanisms easier for real people to understand? The following laboratory experiment provides a straightforward test: we compare pairs of mechanisms that implement the same choice rule. One mechanism in each pair is SP, but not OSP. The other mechanism is OSP. Standard game theory predicts that both mechanisms will produce the same outcome. We are interested in whether subjects play the dominant strategy at higher rates under OSP mechanisms.

Even among theorists. (!!!) Akbarpour, Malladi & Saberi (2020)

Just a Few Seeds More: Value of Network Information for Diffusion*

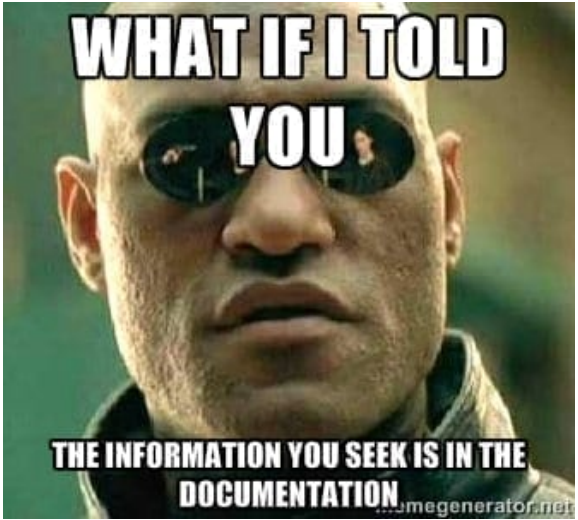
Identifying the optimal set of individuals to first receive information ('seeds') in a social network is a widely-studied question in many settings, such as diffusion of information, spread of microfinance programs, and adoption of new technologies. Numerous studies have proposed various network-centrality based heuristics to choose seeds in a way that is likely to boost diffusion. Here we show that, for the classic SIR model of diffusion and some of its generalizations, randomly seeding $s + x$ individuals can prompt a larger diffusion than optimally targeting the best s individuals, for a small x. We prove our results for large classes of random networks, and verify them in several small, real-world networks. Our results identify practically relevant settings under which collecting and analyzing network data to boost diffusion is not cost-effective.

Why is coding important?

2 General Programming Advice

Stata

Rule # 1


WHAT IF I TOLD YOU
THE INFORMATION YOU SEEK IS IN THE DOCUMENTATION
...memegenerator.net

2 General Programming Advice

Stata / R / MATLAB / Python

- There is no single one-size-fits-all answer. Some problems even require all of them.

Stata / R / MATLAB / Python

- There is no single one-size-fits-all answer. Some problems even require all of them.

	Stata	**R / Python**	**MATLAB**
Best use	Off-the-shelf statistical analysis (e.g., regressions)	Statistical analysis, especially spatial analysis and ML	Matrix based computations (e.g., optimization)

Stata / R / MATLAB / Python

- There is no single one-size-fits-all answer. Some problems even require all of them.

	Stata	**R / Python**	**MATLAB**
Best use	Off-the-shelf statistical analysis (e.g., regressions)	Statistical analysis, especially spatial analysis and ML	Matrix based computations (e.g., optimization)
Programming-type	[✗] Data oriented	[✓] Object oriented	[✓] Object oriented

Stata / R / MATLAB / Python

- There is no single one-size-fits-all answer. Some problems even require all of them.

	Stata	**R / Python**	**MATLAB**
Best use	Off-the-shelf statistical analysis (e.g., regressions)	Statistical analysis, especially spatial analysis and ML	Matrix based computations (e.g., optimization)
Programming-type	[✗] Data oriented	[✓] Object oriented	[✓] Object oriented
Cost	[✗] Proprietary and expensive	[✓] Open source and easy to install in other computers	[✗] Proprietary and expensive

Stata / R / MATLAB / Python

- There is no single one-size-fits-all answer. Some problems even require all of them.

	Stata	**R / Python**	**MATLAB**
Best use	Off-the-shelf statistical analysis (e.g., regressions)	Statistical analysis, especially spatial analysis and ML	Matrix based computations (e.g., optimization)
Programming-type	[✗] Data oriented	[✓] Object oriented	[✓] Object oriented
Cost	[✗] Proprietary and expensive	[✓] Open source and easy to install in other computers	[✗] Proprietary and expensive
Data size	[✗] Restricted by RAM	[✓] SQL implementations allow large (> 100GB) files	[✓] Large matrices and sparse data

Stata / R / MATLAB / Python

- There is no single one-size-fits-all answer. Some problems even require all of them.

	Stata	**R / Python**	**MATLAB**
Best use	Off-the-shelf statistical analysis (e.g., regressions)	Statistical analysis, especially spatial analysis and ML	Matrix based computations (e.g., optimization)
Programming-type	[✗] Data oriented	[✓] Object oriented	[✓] Object oriented
Cost	[✗] Proprietary and expensive	[✓] Open source and easy to install in other computers	[✗] Proprietary and expensive
Data size	[✗] Restricted by RAM	[✓] SQL implementations allow large (> 100GB) files	[✓] Large matrices and sparse data
Easiness to learn	[✓] Very easy to learn	[✗] Steep learning curve and syntax often changes	[✗] Steep learning curve

Stata / R / MATLAB / Python

- There is no single one-size-fits-all answer. Some problems even require all of them.

	Stata	**R / Python**	**MATLAB**
Best use	Off-the-shelf statistical analysis (e.g., regressions)	Statistical analysis, especially spatial analysis and ML	Matrix based computations (e.g., optimization)
Programming-type	[✗] Data oriented	[✓] Object oriented	[✓] Object oriented
Cost	[✗] Proprietary and expensive	[✓] Open source and easy to install in other computers	[✗] Proprietary and expensive
Data size	[✗] Restricted by RAM	[✓] SQL implementations allow large (> 100GB) files	[✓] Large matrices and sparse data
Easiness to learn	[✓] Very easy to learn	[✗] Steep learning curve and syntax often changes	[✗] Steep learning curve
Easiness to upgrade	[✓] Version-based	[✗] Module based	[✓] Version-based

New this year: Python

- Macro sequence traditionally taught with Matlab:
  - programming camp;
  - help from TA;
  - problem set solutions.

New this year: Python

- Macro sequence traditionally taught with Matlab:
  - programming camp;
  - help from TA;
  - problem set solutions.

- **This year** were supporting Python too
  - programming camp web page;
  - TA can help with Python;
  - at least some problem set solutions in Python (others in Matlab).

New this year: Python

- Macro sequence traditionally taught with Matlab:
  - programming camp;
  - help from TA;
  - problem set solutions.

- **This year** were supporting Python too
  - programming camp web page;
  - TA can help with Python;
  - at least some problem set solutions in Python (others in Matlab).

- Why Python
  - Popular!
  - Big community
  - Free and open source
  - General purpose: “second best language for everything”
  - For economists: more uses outside of macro problem sets

Challenge yourself

1. If you only know excel (like I did)... learn Matlab
2. If you know Matlab... learn Python
3. If you know Matlab and Python... learn Julia

- Big benefit to knowing a little bit of many languages

Challenge yourself

1. If you only know excel (like I did)... learn Matlab
2. If you know Matlab... learn Python
3. If you know Matlab and Python... learn Julia

- Big benefit to knowing a little bit of many languages

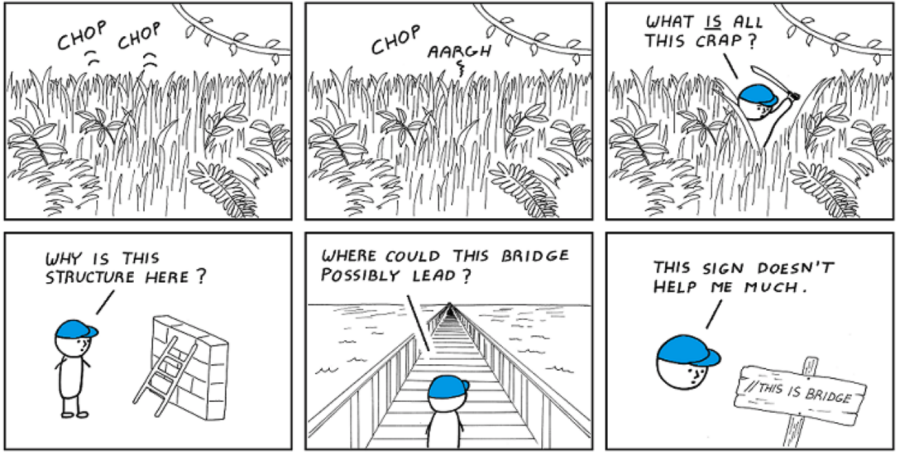
2 General Programming Advice

What do we mean by "good" coding?

1. Code that is correct

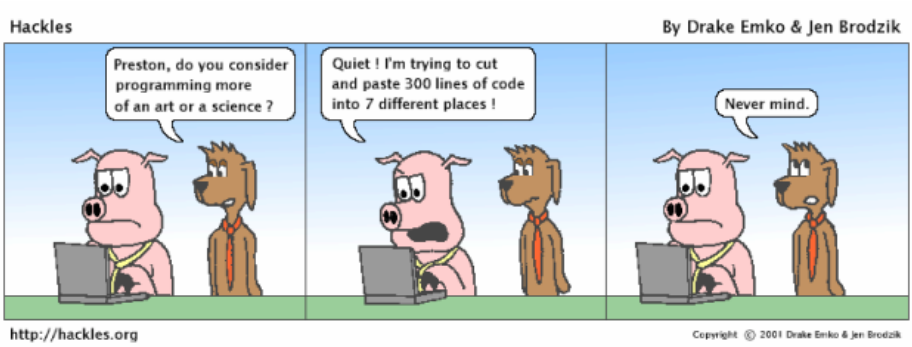
What do we mean by "good" coding?

1. Code that is correct
2. Code that is easy to understand: by you (later on) or by others.



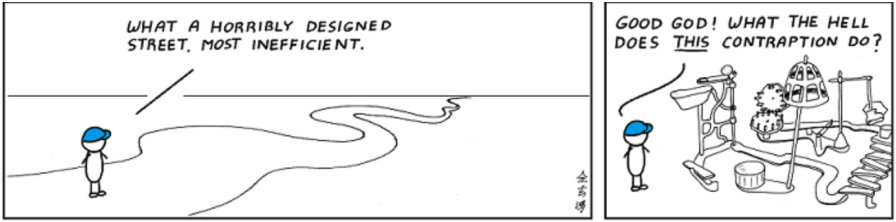
What do we mean by "good" coding?

1. Code that is correct
2. Code that is easy to understand: by you (later on) or by others.
3. Code that is easy to modify.



What do we mean by "good" coding?

1. Code that is correct
2. Code that is easy to understand: by you (later on) or by others.
3. Code that is easy to modify.
4. Code that is efficient and simple to use.



What do we mean by "good" coding?

1. Code that is correct
2. Code that is easy to understand: by you (later on) or by others.
3. Code that is easy to modify.
4. Code that is efficient and simple to use.

   ... and many more!

Commenting your code

- Commenting your code is a useful way to know what's going on.

On Stata:

```
 * One line comment
<code here>

/* Multiline comment
keeps on going here */
<code here>

<code here> // Same line comment
```

On MATLAB:

```
 % One line comment
<code here>

<code here> % Same line comment
```

- But... avoid obvious comments at all costs!

Indenting your code and constructing "snippets”

Good:

```
% INITIALIZE PARAMETERS
flag = 0;
iter = 1;
sum = 0;

% APPROXIMATE SUM OF 1/n^2
while (flag == 0 && iter < max_iter)
    sum = sum + (1/iter)^2;
    iter = iter + 1;
    flag = (1/iter)^2 < tol;
end

% PRINT THE VALUE OF APPROXIMATION
sprintf('pi^2/6 is %0.9f.',sum)
```

Not-so-good:

```
flag = 0;
        iter = 1;

sum = 0;
while (flag == 0 && iter < max_iter)
sum = sum + (1/iter)^2;
iter = iter + 1; % add one


    flag = (1/iter)^2 < tol;
end
sum
```

Variable naming conventions

`STOREID`	`PlaceId`	`place_id`
`HOUSEHOLDINCOME`	`HouseholdIncome`	`household_income`
`PRICE`	`Price`	`price`

- Chose whichever suits you best... but be consistent throughout!

  And whatever you do, don't call your variables `blah, blahh, blahh`.

Multiline functions

- Commands in Stata and MATLAB can get too long to fit in a single line.
- Use /// (Stata) and ... (MATLAB) to separate them into multiple lines.

Good:

```
twoway (scatter mapmt mapmt_pd [w = rw], msymbol(smcircle_hollow) msize(vsmall) mcolor(gs6%40)) ///
  (line mapmt_pd mapmt_pd, lpattern(dash) sort lcolor(black)) ///
  (lowess mapmt mapmt_pd, lwidth(medthick) lcolor(maroon)) ///
  if inrange(mapmt_pd,`=r(p5)',`=r(p95)'), ///
      xtitle("MA payments in 2010 (public data)") ytitle("MA payments in 2010 (private data)") ///
      xlabel(,grid glpattern("..") glcolor(black)) ylabel(,grid glpattern("..") glcolor(black)) ///
      legend(order(3 "lowess" 2 "y = x")) ///

      graphregion(color(white)) plotregion(lcolor(black))
```

Not-so-good:

```
twoway (scatter mapmt mapmt_pd [w = rw], msymbol(smcircle_hollow) msize(vsmall) mcolor(gs6%40)) (line mapmt_pd mapmt_pd, lpattern(dash) sor
```

Batch / Shell script files

- Batch (`.bat` Windows) / Shell (`.sh` Unix) files can help to make your code as automatic as possible.

- Imagine you have a `run_code.sbatch` that inside says:

```
matlab < matlab_code.m > matlab_output_file.txt
...
stata-mp < stata_code.do > stata_output_file.txt
```

  so that by running this file you can run all of your code.

- You may need a few tweaks to make this work on your computer:
  - Add paths to Stata, MATLAB and R.
  - Apply permissions to your files so that they can be run from the command line.

Organizing your project

- We have learnt how to use MATLAB and Stata, and produce `.do` and `.m` files.

- When working on a project, ideally we would want it to *replicate* from beginning to end.

$$\text{raw data} \longrightarrow \text{code} \longrightarrow \text{results}$$

  which most of the times requires:

  1. Cleaning and merging / appending multiple file sources.
  2. Writing descriptive analysis about the data at hand.
  3. Performing estimations (regressions, structural, etc) on the modified data.
  4. Outputting results in clean and concise tables / figures.

- Note that changes in (1) and (3) usually can propagate up until (4). Thus, having portable code is a must.

Organizing your project: Not-so-good

```
---C:/tv_and_potato/---

chips.csv       mergefiles.do         tv_potato_submission.pdf
cleandata.do    regressions_alt.do    tv_potato.tex
extract0B.xls   regressions_alt.log   tv.csv
fig1.eps        regressions.do        tvdata.dta
fig2.eps        regressions.log       rundirectory.bat
figures.do      tables.txt            export_to_csv.stc
```

Organizing your project: Good

```
---C:/build---
/input
    extract0B.xls

/code
    rundirectory.bat
    export_to_csv.stc
    mergefiles.do

/output
    tvdata.dta

/temp
    chips.csv
    tv.csv
```

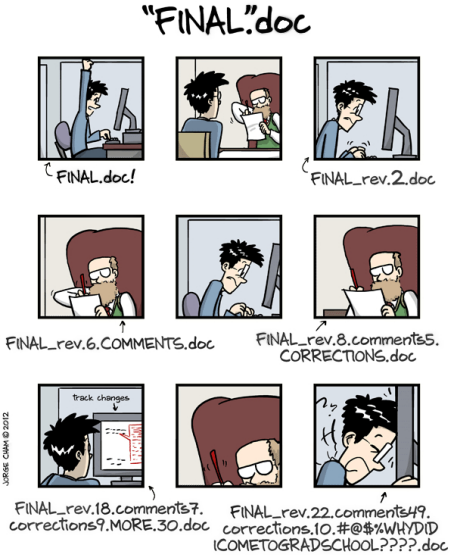
```
---C:/analysis---
/input
    tvdata.dta (link to C:/build/output)

/code
    rundirectory.bat
    regressions.do
    regressions_alt.do

/output
    fig1.eps
    fig2.eps
    tables.txt

/temp
    regressions.log
    regressions_alt.log
```

Organizing your project: Even better if you use numbering

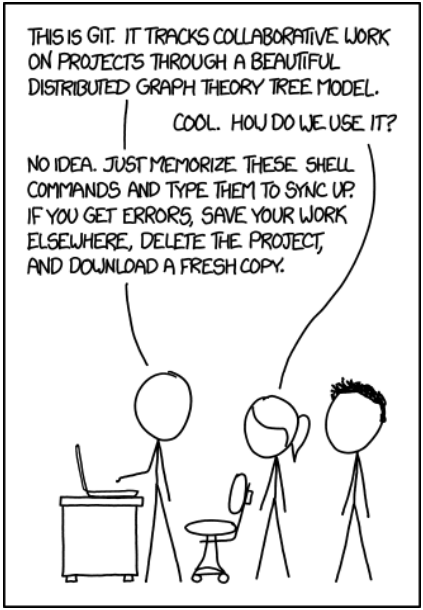
Name	Date modified	Type
0.0_run_file	7/31/2021 10:17 PM	Stata Do-file
0.1_import_ipums	8/5/2021 9:49 AM	Stata Do-file
0.2_import_CleanVariables	8/5/2021 10:08 AM	Stata Do-file
1.1.1_rent_income_ratio	8/24/2021 10:11 PM	Stata Do-file
1.1.2_rent_income_ratio_by_state	8/8/2021 10:22 PM	Stata Do-file
1.2.3_price_income_ratio	8/11/2021 10:42 PM	Stata Do-file
1.2.4_k_means	8/21/2021 2:59 PM	Stata Do-file
1.2.5_ownership_rates	8/24/2021 6:23 PM	Stata Do-file
999.9_temp	8/5/2021 4:07 PM	Stata Do-file

2 General Programming Advice

version2_final_final_ESBcomments_final.pdf

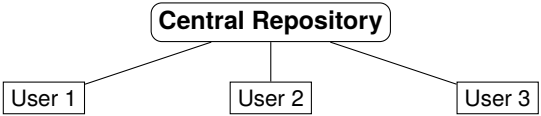


The git


THIS IS GIT. IT TRACKS COLLABORATIVE WORK ON PROJECTS THROUGH A BEAUTIFUL DISTRIBUTED GRAPH THEORY TREE MODEL.
COOL. HOW DO WE USE IT?
NO IDEA. JUST MEMORIZE THESE SHELL COMMANDS AND TYPE THEM TO SYNC UP. IF YOU GET ERRORS, SAVE YOUR WORK ELSEWHERE, DELETE THE PROJECT, AND DOWNLOAD A FRESH COPY.

Using Version Control

- Git is a version control system (think of R). Github is one implementation (think of RStudio) of Git.
- It's how software (and self driving cars!) are made
- Compared to Dropbox:
  - ✓ Github is better tailored to back up code. For example, you can search for specific terms in the history.
  - ✓ Designed to keep a complete history of files, tracking changes, the users who made those changes, etc.
  - ✗ Unlike Dropbox, it does not back up automatically. So it requires effort on the user side.
  - ✗ Not set up for large data. Keep data on Github Large File Storage or Dropbox



- The central repository contains the code and complete history. Usually located in Github or BitBucket.
- Users can "push" (e.g., send) changes to central repository, or "pull" (e.g., receive) changes made by others.

armenia_cba / savings / armenia_savings

Commits

Clone

Search commits

All branches

Author	Commit	Message	Date
Josh Kim	d176338	Update paper with Vahagn edits	2020-05-21
Josh Kim	8c737d5	Compile MLA style references page	2020-05-04
Diego Jimenez Hernan...	6fffe5f	Chicago bib	2020-05-03
Josh Kim	c762fee	Minor edits to conclusion	2020-05-03
Josh Kim	f2877b6	Finish edits to results section	2020-05-03
Josh Kim	8aae436	Edits to background and empirical specification sections	2020-05-03
Josh Kim	9383ba3	Begin background edits	2020-05-03
Josh Kim	5887fe2	Finish intro edits	2020-05-03
Josh Kim	44eab17	Recompile with new bibliography	2020-05-03
Alik Shirkhanyan	ae4bb63	Deposits file to show HH role in dollarization, the library.bib changed	2020-05-03
Josh Kim	6dc7e9e	More intro edits	2020-05-02
Josh Kim	c0057ee	Intro rewrites	2020-05-02
Josh Kim	101cdde	Add all references we want	2020-05-02
Alik Shirkhanyan	ce862ad	small change in preamble, modified lit review in introduction	2020-05-02
Josh Kim	d3644da	Add bandwidth estimates	2020-04-26

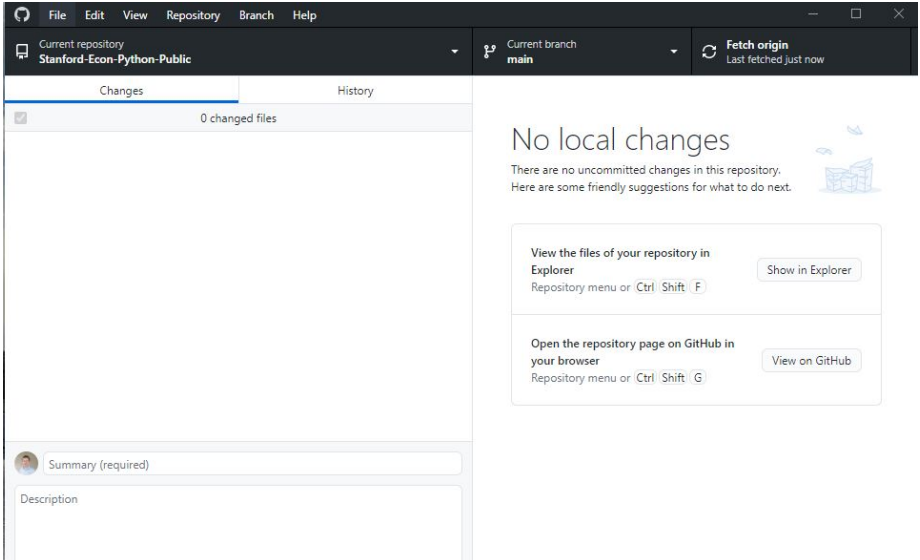
Paper/introduction.tex MODIFIED

Side-by-side diff | View file | Comment | •••

```diff
 % !TeX root = paper.tex
+\newpage
 \section{Introduction}
 \label{sec:intro}

-Many people live in developing economies where the value of their local currency is unstable and can depreciate unexpectedly. This instabi
+Many people live in developing economies where the value of their local currency is unstable and can depreciate unexpectedly. This instabi

 At the macro level, financial dollarization has a number of negative effects. First, dollarization increases the risk of financial crisis;

-At the micro level, financial dollarization largely affects the ability of individuals to safely save money. To do so, individuals must ma
+At the micro level, financial dollarization introduces a series of complex decisions that may affect the ability of individuals to save. T

-In this paper, we study how currency depreciations affect individual savings decisions. To do so, we exploit three details of the Armenian
+In this paper, we study how currency depreciations affect individual savings decisions. To do so, we use detailed savings data from a larg
+
+
+To do so, we exploit three details of the Armenian financial system. The first is that Armenia is a moderately dollarized economy where in

 To study individual savings decisions, we use detailed savings data from a large Armenian commercial bank. We can then track the savings p
```

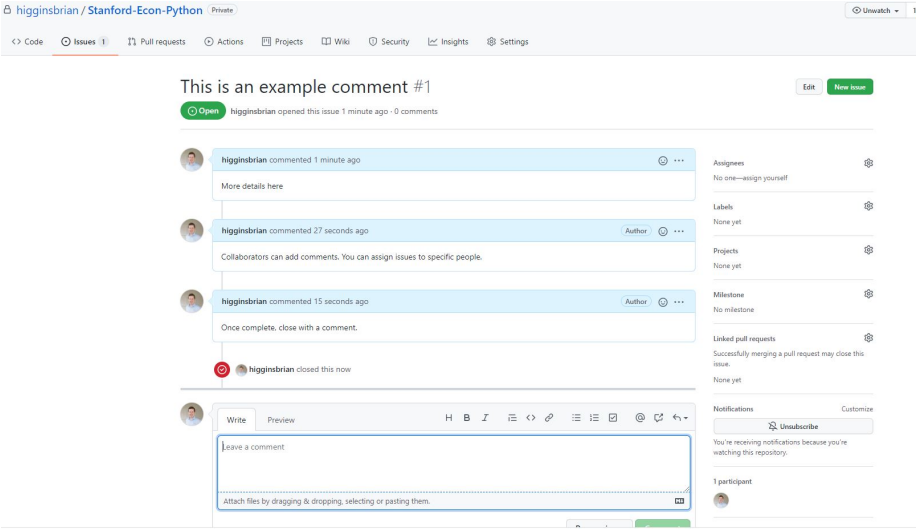
Other useful features

- Github Desktop



Other useful features

- Github Desktop
- Track issues



Other useful features

- Github Desktop
- Track issues
- Turn a repo into a webpage using GitHub Pages

Python for Stanford Economics PhD students

This is a repo for Stanford Economics students who want to use Python during the first year PhD sequence.

Why Python?

1. **Popular**: one of the most used programming langugages in the world.
2. **Community**: the python community provides lots resources to make python better: open source packages; and forums with plenty of questions and answers; and videos and tutorials to use packages.
3. **Free and open source**: You can always look under the hood to see what packages doing. Anyone can run your code without paying for software like Matlab or Stata, and Jupyter notebooks make it easy to share your code and results together.
4. **General purpose**: Modern economic research involves lots of tasks, many of which can be done in python: computing and estimating models (numpy, scipy, numba), data analysis (pandas), doing algebra (sympy), mapping (geopandas), machine learning (keras, tensorflow, pytorch), webscraping (beautiful soup), text analysis (nlkt), digitizing records (layout-parser), creating websites (Jupyter plus GitHubPages), parallel programming on the GPU (cupy, numba).

What we do

In this repo we provide resources to learn python and get ready for the first year sequence. We'll let you know when we add more material.

1. Getting set up with Python [link]
2. Starting to program with Python [Notebook] Open in Colab

Other useful features

- Github Desktop
- Track issues
- Turn a repo into a webpage using GitHub Pages
- Add paths or file types to *.gitignore* text file
  - Git will not transfer these to the common repository
  - /*data* folder
  - latex intermediate output files: .nav, .run, .toc, .aux

Using Github

- Excellent news! Stanford students have access to Github Pro, which allows for private repositories.

Using Github

- Excellent news! Stanford students have access to Github Pro, which allows for private repositories.
- Main commands to use on git (type in the terminal):

Using Github

- Excellent news! Stanford students have access to Github Pro, which allows for private repositories.
- Main commands to use on git (type in the terminal):
  - `git clone [URL]` “clones” the information in the central repository to your computer — the project could be empty!

Using Github

- Excellent news! Stanford students have access to Github Pro, which allows for private repositories.
- Main commands to use on git (type in the terminal):
  - `git clone [URL]` "clones" the information in the central repository to your computer — the project could be empty!
  - `git status` provides information on the local changes, relative to the last commit.

```
(base) djavierjh@DNa819b7e MilkProjectPaper % git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   Figures/distance_regressions_any_purchase.pdf
        modified:   Figures/distance_regressions_price_liter.pdf
        modified:   Figures/distance_regressions_purchase_liters.pdf
        modified:   Sections/appendix_estimation.tex
        modified:   Sections/figures.tex
        modified:   Sections/section_model_theory.tex
        modified:   preamble.tex

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Figures/map_distributionLALA.pdf
        Figures/map_distributionLECHERAGUADALAJARA.pdf

no changes added to commit (use "git add" and/or "git commit -a")
```

Using Github

- Excellent news! Stanford students have access to Github Pro, which allows for private repositories.

- Main commands to use on git (type in the terminal):
  - `git clone [URL]` “clones” the information in the central repository to your computer — the project could be empty!
  - `git status` provides information on the local changes, relative to the last commit.
  - `git pull` downloads the new changes from the central repository since your last pull

Using Github

- Excellent news! Stanford students have access to Github Pro, which allows for private repositories.

- Main commands to use on git (type in the terminal):
  - `git clone [URL]` “clones” the information in the central repository to your computer — the project could be empty!
  - `git status` provides information on the local changes, relative to the last commit.
  - `git pull` downloads the new changes from the central repository since your last pull
  - `git add [files]` is used to mark what files do you want from your local version to send to the central repository

Using Github

- Excellent news! Stanford students have access to Github Pro, which allows for private repositories.

- Main commands to use on git (type in the terminal):
  - `git clone [URL]` “clones” the information in the central repository to your computer — the project could be empty!
  - `git status` provides information on the local changes, relative to the last commit.
  - `git pull` downloads the new changes from the central repository since your last pull
  - `git add [files]` is used to mark what files do you want from your local version to send to the central repository
  - `git commit -m "message here"` updates takes the files marked by “add” and produces a local copy

Using Github

- Excellent news! Stanford students have access to Github Pro, which allows for private repositories.

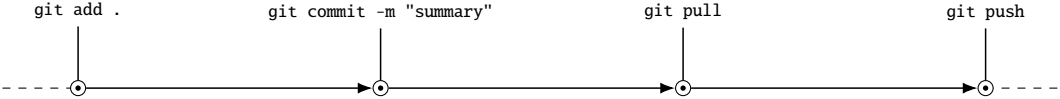
- Main commands to use on git (type in the terminal):
  - `git clone [URL]` “clones” the information in the central repository to your computer — the project could be empty!
  - `git status` provides information on the local changes, relative to the last commit.
  - `git pull` downloads the new changes from the central repository since your last pull
  - `git add [files]` is used to mark what files do you want from your local version to send to the central repository
  - `git commit -m "message here"` updates takes the files marked by “add” and produces a local copy
  - `git push` uploads your changes marked by commits to the central repository

Using Github

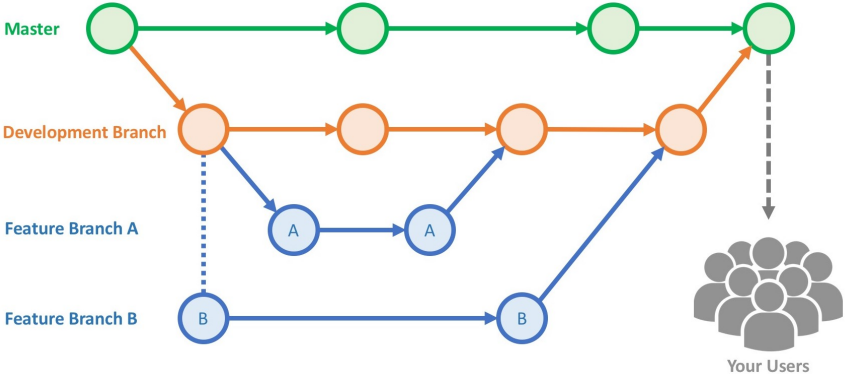
- Excellent news! Stanford students have access to Github Pro, which allows for private repositories.

- Main commands to use on git (type in the terminal):
  - `git clone [URL]` “clones” the information in the central repository to your computer — the project could be empty!
  - `git status` provides information on the local changes, relative to the last commit.
  - `git pull` downloads the new changes from the central repository since your last pull
  - `git add [files]` is used to mark what files do you want from your local version to send to the central repository
  - `git commit -m "message here"` updates takes the files marked by “add” and produces a local copy
  - `git push` uploads your changes marked by commits to the central repository

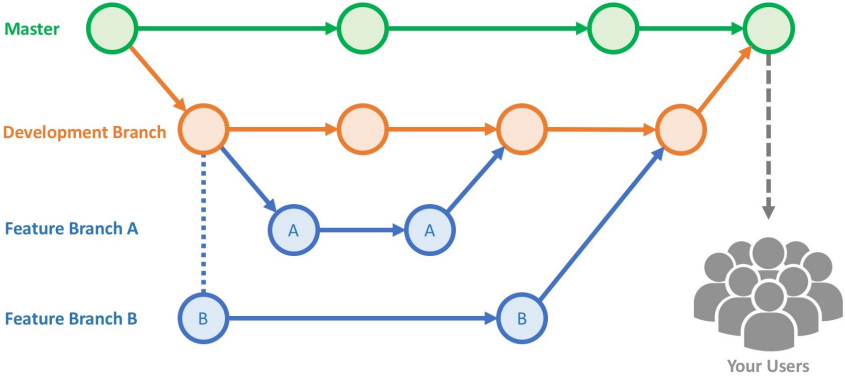
- One key thing to remember:



Github branching



Github branching



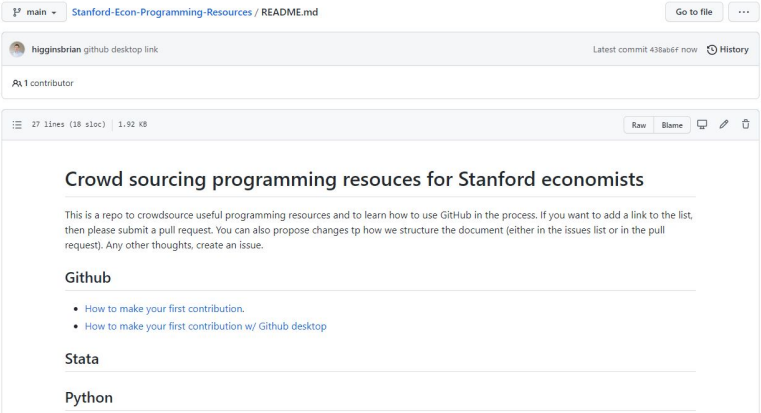
1. Create a new branch called *dev* (or a fork, if it is someone elses repo)
2. Pull, add, commit and push to this branch
3. When ready, submit a **pull request** to merge *dev* into the *main*
   - Pull requests are reviewed before merging (by you, collaborators, or owner of the repo)

Challenge yourself

1. Sign up to Github and create your first repo

Challenge yourself

1. Sign up to Github and create your first repo
2. Submit your first *Pull request* to `https://github.com/higginsbrian/Stanford-Econ-Programming-Resources`
   1. Fork this repo to your own account
   2. Add a link, then commit and push changes to your own forked repo (on Github or locally on your computer)
   3. Submit a pull request to merge your edits into my repo



Challenge yourself

1. Sign up to Github and create your first repo
2. Submit your first *Pull request* to `https://github.com/higginsbrian/Stanford-Econ-Programming-Resources`
3. Clone our Python repo and start learning python
4. Use git to track your macro problem sets
5. Create a web page with Github Pages

Some other useful tools

1. **Makes files:** reproduce entire project start-to-finish
2. **Notebooks:** Jupyter (Python, Julia, R, Stata); R markdown; Matlab live editor
3. **Integrated computing environments (IDEs)**: debugging, code completion, github support

Why is coding important?

General Programming Advice

3 Stata

Global and local variables

local macros `‘macro’`	Macro that cannot be accessed out of the program.
global macros `$macro`	Macro that can be accessed out of the program.

Examples:

```
local loss = 0.5
generate income = revenue - ‘loss’*costs


global path = "Users/username/Desktop/"
...
use $path/data, replace
```

Both local and global macros can be nested. More on this on 7.

For loops / If conditional statements

`forvalues`	Loop over numerical values.
`foreach`	Looper over variables. Can be used with locals and variable lists.
`if`	Check if conditions.

Example: Compare these two code snippets.

```
local spec1 ""
local spec2 "age agesq educ"
local spec3 "`spec2' mo_educ fa_educ"


foreach sampleCond in "if male" "if !male" "" {
  forvalues = 1/3 {
    reg y x `spec`i'' `sampleCond'
  }
}
```

```
reg y x if male == 1
reg y x age agesq educ if male == 1
reg y x age agesq educ mo_educ fa_educ if male
reg y x if male == 0
reg y x age agesq educ if male == 0
reg y x age agesq educ mo_educ fa_educ if !male
reg y x reg y x age agesq educ
reg y x age agesq educ mo_educ fa_educ
```

Say that you want to add another variable. Which one is easier to modify?

Programs in Stata

- Programs are a useful way to generate your own commands. For example:

```
capture program drop _all
program output_regressions
syntax varlist, [absorb(string)] [control(string)] depvar(string)

eststo clear
quietly {
  foreach var of varlist ‘varlist’ {
    eststo:  reghdfe ‘var’ ‘depvar’ ‘control’ [pw = wt], absorb(‘absorb’) cluster(id)
  }
}

estout, cells(b(fmt(%04.3f)) se(par) _star) keep(‘depvar’) ///
   starlevels(* 0.1 ** 0.05 *** 0.01) ///
   stat(N r2, fmt(%03.2f %03.2f %8.0fc %04.3f) label("observations" "r-squared")) ///
   numbers mlabels(,depvars) style(tex)
end

 output_regressions price_var1 price_var2 price_var3, ///
   absorb(county#month) control(control1 control2) depvar(devpvar1)
```