# What Economists Can Learn from Programmers (and Vice Versa!)

# This isn't an econ conference!

True, but...

- Economists do a lot of programming!
- The programs often have serious policy implications!
- Tech and econ cultures have a lot of similarities!
- Some parts of the econ philosophy have changed my views on programming, and vice versa!

# A Bit about Me

- Carnegie Mellon (CS, 2 years, left)
- Employment
  - Business analysis
  - IT consulting
  - System administration
- CMU (Economics, 2 years PT, 2 years FT)
- Northwestern (Economics PhD, 2 years, left)
- Solutions for Progress
  - Research Assistant
  - Software Engineer
- Comcast
  - DevOps Engineer

# What Economists Can Learn from Programmers

# Sustainable Development!

# Modern Econ Is Programming!

- Statistical modeling is the main route for empirical work
- Computational econ is an increasingly popular topic
  - Simulating economic agents using AI
- Availability of GPGPUs makes large-scale optimization problems faster and cheaper to solve
- Algorithmic game theory is of considerable interest in both CS and economics
  - Auctions, load balancing, routing

# Coding Gets Short Shrift

- Time pressure of classes
- No formal training for development
  - Code is part of homework assignments, but little direction or assistance
- Using code written by a handful of people sans documentation
- Programming/data munging/debugging often left to harried research assistants
  - What happens when *you're* that RA?
- No time to learn Fortran, no time to write it in Python
- If you're unlucky, you might only get an executable

# A few years later...


(Credit: theprofoundprogrammer.com)

**…well, I *hope* it works!**

# But There's Hope!

- quant-econ project (open source Python, Julia libraries with tests!)
- Software Carpentry (general skills)
- FRED (Fed Reserve Economic Data), Quandl
  - Standardized APIs for economic data
- Open Knowledge Foundation

# Algorithms!

# Don't need to be an algorithms expert, but...

- Good to have a grasp on asymptotic behavior
  - Datasets are often bigger than you expect (but smaller than you hope)
  - Bonus: you get to use some of the ideas you might've picked up in econometrics!
- If you find yourself with nested loops, take a step back
  - You might be duplicating calculations
- Don't do more than you have to!
  - Memoization, dynamic programming

# Testing

# You've got results!

But...

- Are they consistent?
- Did you make sure you have your controls?
- You made a complicated model -- did you check to make sure that its implications are what you thought?
- You're using your colleague's libraries, but their grasp of IEEE-754 floating point is incomplete

# How Bad Could It Be?

# Disasters in Econ Research

- Donohue and Levitt (2001) (the "abortions and crime" paper)
  - Code neglected to include controls
  - Corrected, impact nearly zero
- Feldstein (1974) (Social Security depresses savings)
- Reinhart and Rogoff (2010) (High debt reduces growth)
  - Significant policy implications re: EU austerity
  - Copy-paste errors overstated relationship

# What Can We Do?

- Learn to write tests
  - Unit tests for smaller model components
  - Integration tests for full models
  - Regression (no pun intended) tests
- Standard libraries
- Explicit checklists for control variables
- GitHub (or GitLab)!
  - Or some other collaborative system
  - Also get version control

# What Programmers Can Learn from Economists

# Causation


Caroline, via econlolcats.tumblr.com

# Modeling in One Slide

- We want to evaluate how much impact some factors *X* has on a variable of interest *Y*
- We build models, most frequently regressions, to numerically estimate these effects
- A core assumption is that *X* is uncorrelated with *Y*
  - When this doesn't happen, we have to deal with *endogeneity bias*
- The most popular way to deal with that is replace the *X* with a variable *Z* that's both correlated with *X but not with Y*, called an instrument
  - We then run the regression with the instrument in place of *X*.

# Troubleshooting Like an Economist

When fixing technical problems...

- Endogeneity bias everywhere!
  - Even more difficult to avoid in tech debugging (a good instrument is often hard to find)
- Don't ignore institutional effects!
  - Sometimes seemingly technical causes are actually policy-related
- In tech and economic modeling, it's important to understand cultural/historical context
  - Econ models make implicit and explicit assumptions about culture and history
  - Programs we write also make these assumptions (beyond i18n/l10n)

# Specialization (and trade!)

# Generalists are awesome! (but...)

- Life's too short to do everything
  - Do you *really* want to write your own network stack for production?
- Even a skilled generalist can benefit from specializing
  - Novices can help too!
- Specializing is less useful without trade
  - If no one has a use for your COBOL expertise, there's little benefit in focusing on it

# Comparative Advantage!

# David Ricardo

(Portrait by Thomas Phillips)

# Comparative Advantage!

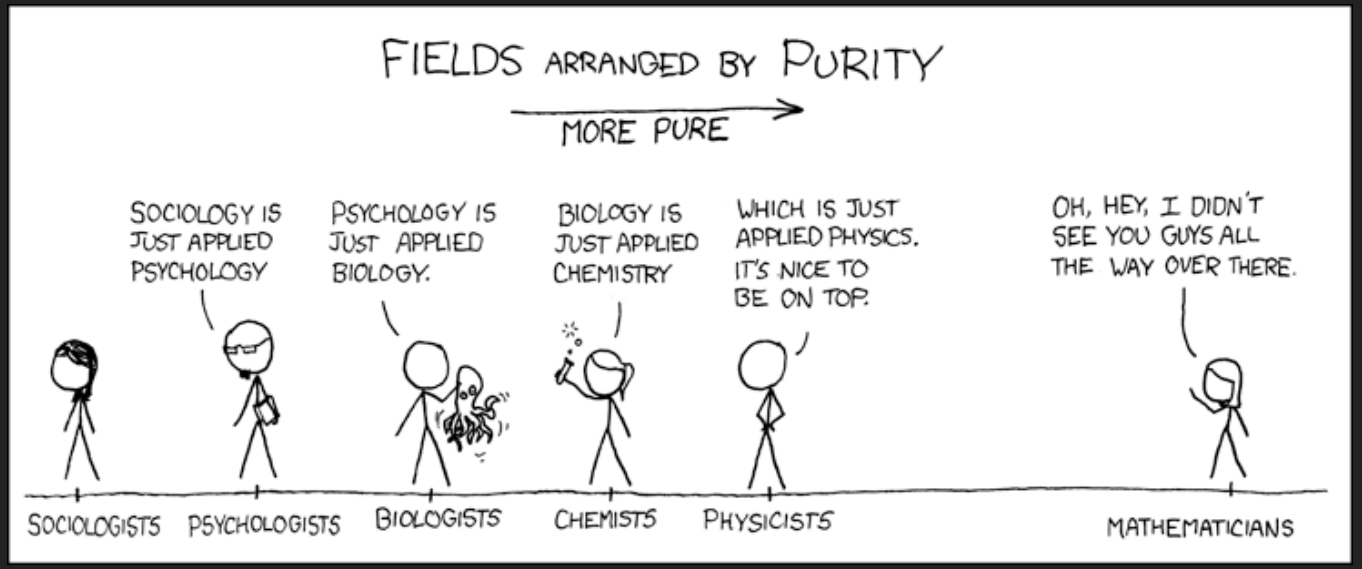
- Absolute advantage: Alice is better at writing network code than Barb; Barb is better at building numerical models
  - Hence, they should do what they do best
- Comparative Advantage: Alice is better at *both* network code and numerical models than Barb, but Alice is better at numerical models
  - Alice should take on the numerical models
  - Barb writes the network code
- Since they can work at the same time, they can produce more than either alone

# Caution: Hasty Generalization!

Note that comparative advantage demonstrates what's *possible*, not necessarily what's *best*

# What *Both* Economists and Developers Can Learn

# Community Reflection and Action


FIELDS ARRANGED BY PURITY
MORE PURE
SOCIOLOGY IS JUST APPLIED PSYCHOLOGY
PSYCHOLOGY IS JUST APPLIED BIOLOGY.
BIOLOGY IS JUST APPLIED CHEMISTRY
WHICH IS JUST APPLIED PHYSICS. IT'S NICE TO BE ON TOP.
OH, HEY, I DIDN'T SEE YOU GUYS ALL THE WAY OVER THERE.
SOCIOLOGISTS
PSYCHOLOGISTS
BIOLOGISTS
CHEMISTS
PHYSICISTS
MATHEMATICIANS

# A Premium on the "Hardcore"

- Both tech and econ spend a lot of time on hierarchies and establishing their fields as "real".
  - Tech: OS hackers are "hardcore/real"; web devs are "not"
  - Econ: Theorists are "hardcore/real"; economic historians are "not"
- Both have forums that advance these ideas
  - Tech: Hacker News
  - Econ: Econ Job Market Rumors

# Only as Good as What We Tolerate

- We need to critically examine our fields
  - "real programmers" and "real economists"
  - Econ: privileging theory over what actually happens
  - Tech: privileging tools over people (both devs and end-users)
- Minorities in both fields are underrepresented and discouraged
  - I was the *only* active AA in my grad department (student or faculty)
  - Racism, sexism, homophobia, transphobia rampant

# What Can We Do?

- More inclusive events
  - Not just across race/gender/class, but also skills and interests
  - The econ equivalent of !!Con?
- Make it easier to get into technical and economic discourse
  - Blogs and Twitter have made a significant shift in voices in econ, both in and out of academia
- Be patient and listen
  - As Catt Small noted, refrain from approaching a problem tools-first, whether economic or programming

In Memoriam

Tiago Pires (1984-2016)