

**Respy**

July 25, 2018

# What's In It for You?

- ▶ You can use it as a learning tool
- ▶ You can use it for your own work
- ▶ It can save you months of work!
  - ▶ Clean and robust solutions to many problems you find in any serious econometric project
  - ▶ Nice interface, refined over years

# Outline

- ① What is Respy?
- ② Quick Tutorial
- ③ Software Engineering

**What is Respy?**

## What Is Respy?

- ▶ Python program to estimate and simulate a structural model of labor market choices and human capital accumulation
- ▶ Under the hood: High performance Fortran code
- ▶ Models are easy to specify, without touching the code
- ▶ Well tested and used for several papers

# Why Is There No Stata Command for Life-Cycle Models?

- ▶ Stata is too slow
- ▶ Stata syntax makes it hard to specify complicated models
- ▶ Every project is different and requires specific computational tricks to become feasible
  - ▶ You will have to program yourself!
- ▶ But you don't have to reinvent the wheel
- ▶ Respy can be a good starting point

# Why Is Simulation So Important?

- ▶ Learn about the model
- ▶ Verify correctness of the code
- ▶ Simulation based estimation methods
- ▶ One of the main goals of structural estimation is simulation of counterfactual policies

# Where to Find Respy

## Source Code:

- ▶ <https://github.com/OpenSourceEconomics/respy/tree/master/respy>

## Documentation:

- ▶ <https://respy.readthedocs.io/en/latest/>



# Quick Tutorial



Move fast and fix things!

Resolve production errors quickly, and deploy code with confidence. Give Rollbar a try.

Sponsored - Ads served ethically

# Welcome to respy's documentation!

[PyPI](#) | [GitHub](#) | [Issues](#)

`respy` is an open-source Python package for the simulation and estimation of a prototypical finite-horizon discrete choice dynamic programming model. We build on the baseline model presented in:

Keane, M. P. and Wolpin, K. I. (1994). [The Solution and Estimation of Discrete Choice Dynamic Programming Models by Simulation and Interpolation: Monte Carlo Evidence](#). *The Review of Economics and Statistics*, 76(4): 648-672.

license [MIT](#)

## Contents:

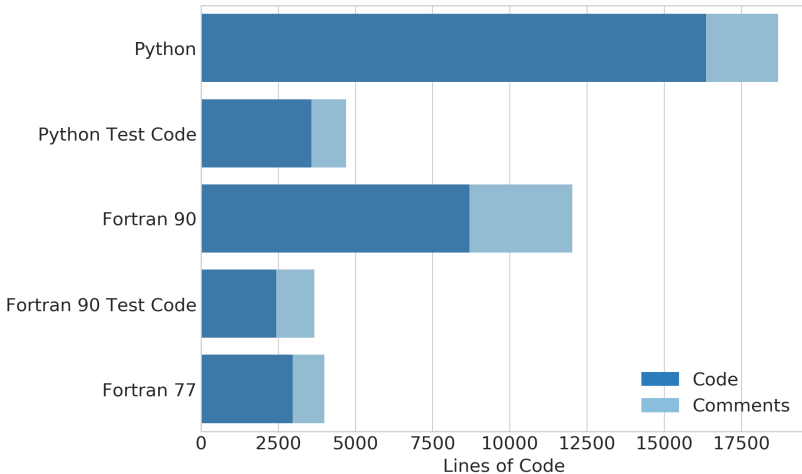
- [Background](#)
- [Installation](#)
- [Setup](#)
- [Tutorial](#)
- [Numerical Methods](#)
- [Reliability](#)
- [Scalability](#)
- [Software Engineering](#)
- [Contributing](#)
- [Additional Details](#)
- [Contact and Credits](#)
- [Changes](#)
- [Bibliography](#)

Next

# **Software Engineering**

# Size of Respy

Lines of Code by Language



# What is Software Engineering?

- ▶ Strategies to handle complexity and avoid bugs in large software projects
- ▶ You don't need it if your project consists of one short do-file
- ▶ Below, I'll mention two principles; you can find more in the documentation

# Testing

- ▶ Respy has thousands of lines of code
- ▶ It's easy to introduce a bug ...  
... and hard to find it afterwards!
- ▶ That's why we have tests at different levels
  - ▶ Regression tests
  - ▶ Integration tests
  - ▶ Unit tests
  - ▶ ...

# Modularity

- ▶ Respy uses pre-existing code for many tasks
  - ▶ Numerical optimization
  - ▶ Numerical integration
  - ▶ Linear algebra
- ▶ Advantages
  - ▶ Less code to maintain
  - ▶ Highly optimized routines
  - ▶ Easy to switch out parts

## Modularity II

- ▶ The switching out part is extremely important!
  - ▶ Projects evolve and goals change
  - ▶ At the beginning it's unknown where changes will be
- ▶ Isolate code for each task even for the code you write yourself!
  - ▶ Solution, Estimation, . . .
- ▶ A well written paper is also modular!