Technology Adoption in Input-Output Networks

Xintong Han[1] Lei Xu[2]

[1]Concordia University

[2]Bank of Canada

Overview

- Input-Output Networks
  - Modern economy: Disaggregation, specialization
  - Propagation of decisions and shocks
  - Policy implication

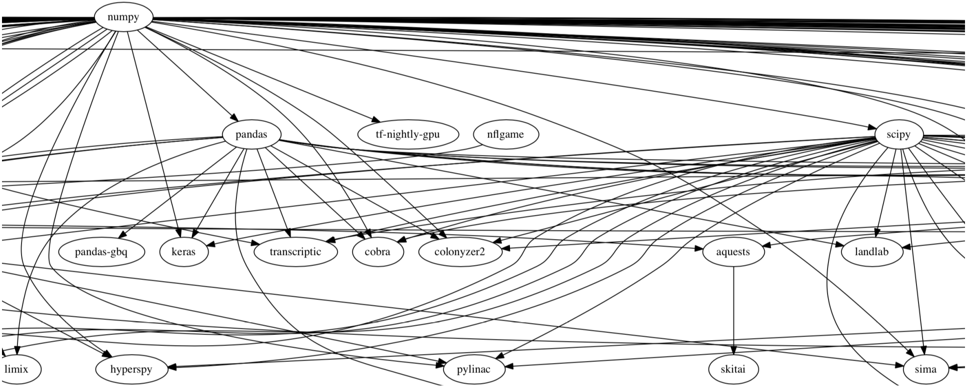
Overview

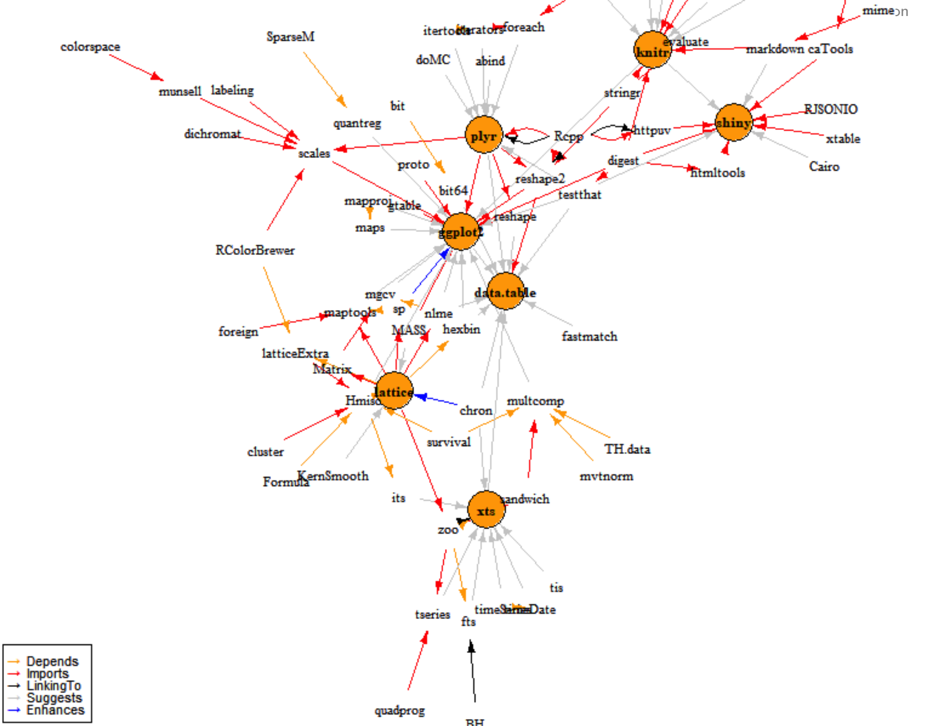
- Input-Output Networks
  - Modern economy: Disaggregation, specialization
  - Propagation of decisions and shocks
  - Policy implication
- Technology Adoption
  - How does innovation propogate through a network?
  - How technology adoption is affected by the network structure?
  - What policies can be implemented to promote technology adoption?

Overview

- Input-Output Networks
  - Modern economy: Disaggregation, specialization
  - Propagation of decisions and shocks
  - Policy implication
- Technology Adoption
  - How does innovation propogate through a network?
  - How technology adoption is affected by the network structure?
  - What policies can be implemented to promote technology adoption?
- Context: the Python programming language
  - Transition from Python 2 to Python 3

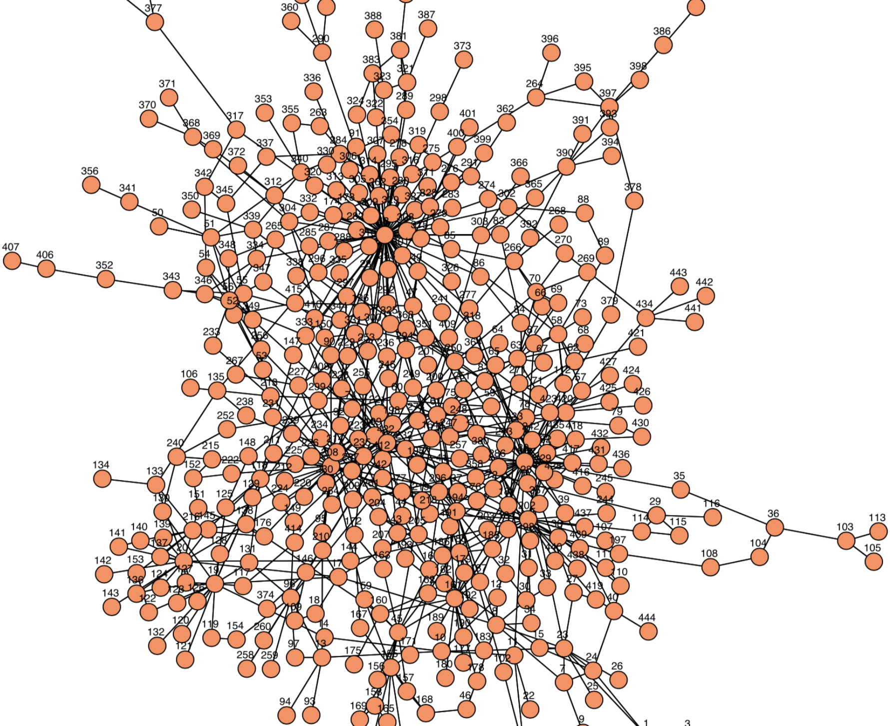
Input-Output Network of Python Packages





colorspace
SparseM
itertools
iterators
foreach
mime
knitr
evaluate
markdown
caTools
doMC
abind
munsell
labeling
stringr
bit
quantreg
plyr
Rcpp
httpuv
shiny
RJSONIO
xtable
dichromat
scales
digest
Cairo
proto
htmltools
reshape2
bit64
testthat
mapproj
gtable
reshape
maps
ggplot2
RColorBrewer
data.table
mgcv
maptools
sp
nlme
foreign
MASS
hexbin
fastmatch
latticeExtra
Matrix
lattice
Hmisc
chron
multcomp
cluster
survival
TH.data
Formula
KernSmooth
mvtnorm
its
sandwich
xts
zoo
tis
tseries
timeSeries
timeDate
fts
quadprog
BH
Depends
Imports
LinkingTo
Suggests
Enhances

In
O



on

Literature & Contribution

Technology Adoption

- Network effects: Mansfield (1968), Katz and Shapiro (1985), Saloner and Shepard (1995)
- Other channels: Gowrisankaran and Stavins (2004), Atkin, Chaudhry, Chaudry, Khandelwal, Verhoogen (2017), etc.

Input-Output / Social Networks
Jackson (2010), Acemoglu (2012), etc.

Technology Adoption with Networks

- Ryan and Tucker (2012): video-calling technology
- Bjorkegren (2018): mobile phones in Rwanda

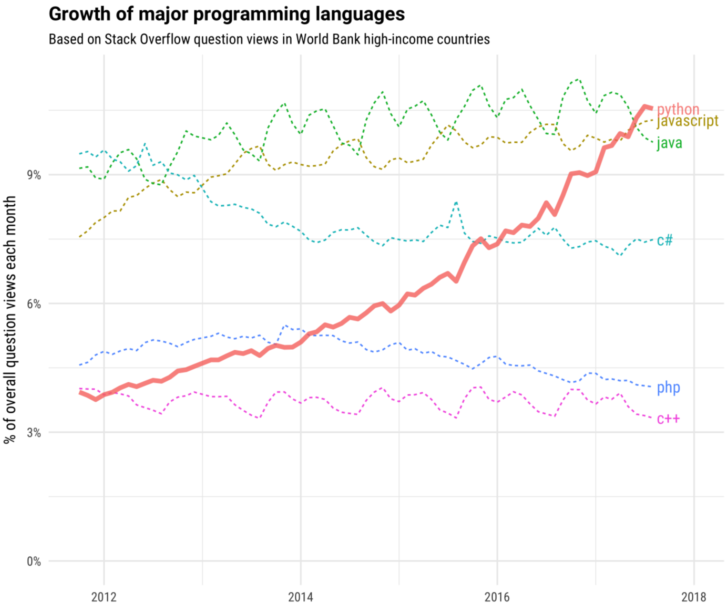
Contribution

- Dynamic model of technology adoption which incorporates an input-output network
- Structural model of network effects
- Demonstrate a new channel that affects technology adoption: the input-output network slows down the speed of adoption for 1.5 years
- Unique universal dataset of Python packages
- Counterfactual analysis of new technology promotion policies (Katz and Shapiro (1985))

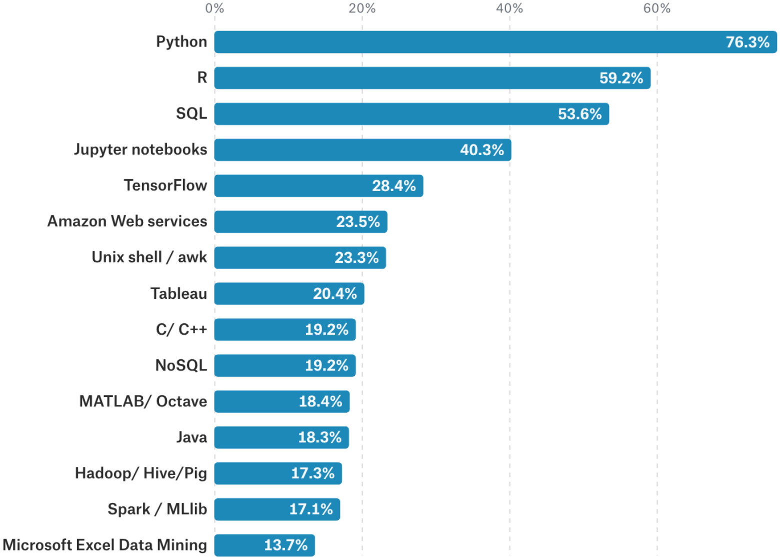
What is Python?

- A general-purpose programming language
  - like C or Java
  - vs. domain-specific languages: Stata, Matlab
  - Third-party packages: >140,000 available
  - Almost all are open source software (OSS)
- One of the most popular programming languages in the world
  - Good choice as a first programming language

Python Popularity



Python Popularity



Technology Adoption: Python 2 to Python 3

- Python 2: 2000
- Python 3: 2008
- New features that require fundamental changes
- Not backward compatible $\longrightarrow$ adoption/switching cost

Python 3 New/Incompatible Features

- Default Encoding System
  - Python 2: ASCII
    - e.g. "café" $\longrightarrow$ error
    - solution: unicode("café", encoding='utf8')
  - Python 3: Unicode
    - e.g. "café" $\longrightarrow$ café
- Print Function
  - Python 2: print "Hello World"
  - Python 3: print("Hello World")
- Division
  - Python 2: $5/2 = 2$
  - Python 3: $5/2 = 2.5$

why incompatible?

Terminology - Packages

- Packages: A collection of tools that enables users to do advanced tasks.
- other names: libraries, modules, (sub)routine

e.g. Matrix Multiplication

$$A = [1, 2, 3], \qquad B = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$$

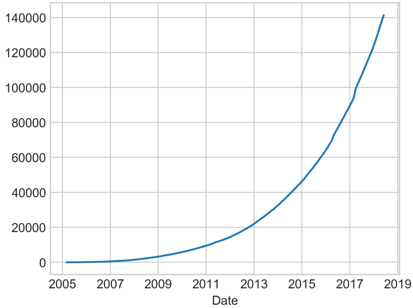
Pure Python - loop

```
A = [1,2,3]
B = [1,1,0]
result = 0
for i in range(3):
    result = result + A[i] * B[i]
```

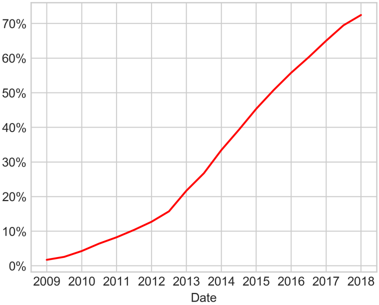
with package NumPy

```
import numpy
A = numpy.array([[1,2,3]])
B = numpy.array([1,1,0])
A @ B
```

Python Packages with Python 3 Support



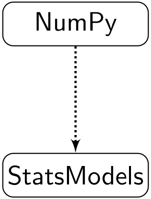
(a) Total Number of Packages



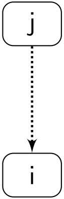
(b) % of Packages with Py3 Support

Input-Output Network - Dependencies

- Division of labor: packages usually have specialties
- e.g. NumPy: linear algebra, numerical analysis, etc
  - $\hat{\beta} = (X'X)^{-1}X'y$
- StatsModels: OLS, GLM, MLE, GMM, etc
  - it requires matrix inversion & multiplications from NumPy
  - i.e. NumPy is StatsModels' dependency



Adoption Cost - Dependencies



Python 2 only

wants to adopt Python 3

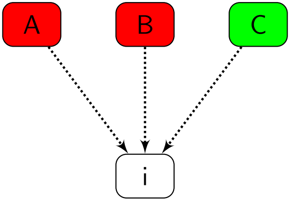
Adoption Costs

1. update one's own codebase
2. dependencies without Python 3 support
   - find an alternative dependency that support Python 3
   - change the required code by oneself

Python 3 Adoption Decision

- $d_{i,t} \in \{0,1\}$: package i's decision to add Python 3 support.
  i.e. support both Python 2 and Python 3
- Irreversible decision
- Each package is considered as an independent agent in the adoption decision

Adoption Cost - Dependencies



Assumption 1 (sequential move): At time t, a package i observes Python 3 adoption decisions made by its dependencies, namely, $d_{j,t} \forall j \in U_{i,t}$.

- $C_{i,t} = AC_0 + \alpha^{size} \cdot Size_{i,t} + \alpha^{\mu} \cdot \mu_{i,t}$

- $U_i = \{A, B, C\}$
- $d_{A,t} = 0$

  $d_{B,t} = 0$

  $d_{C,t} = 1$

- $\mu_{i,t} = \sum_{j \in \mathcal{U}_i} \mathbf{1}\{d_{j,t} = 0\}$

  number of dependencies without Python 3 support.

  in this example, $\mu_{i,t} = 2$

Utility Function - Downloads

- Motivations of OSS Contribution: altruism, ego gratification, career concerns
- Model utility as a function of user downloads
- Denote $x_{i,t} = log(Downloads_{i,t})$
- Time-Varying AR1 Process (package i's belief of future downloads): more info

$$x_{i,t} = \rho_0 + \rho_r \cdot d_{i,t} \cdot r_t + \rho_1 \cdot x_{i,t-1} + \epsilon_{i,t}$$

where

$d_{i,t} \in \{0, 1\}$: Package i's adoption status

r_t: Python 3 adoption rates by packages

current version: perfect foresight

- Endogeneity of $d_{i,t}$: IV

Model - Flow Utility

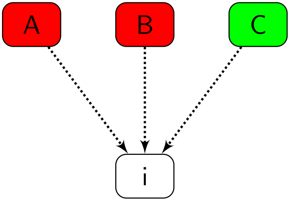
$$u_{i,t} = \alpha^{x} \cdot x_{i,t}(d) - C_{i,t} + \nu^{d}_{i,t}$$

where

$$C_{i,t} = \begin{cases} AC_0 + \alpha^{size} \cdot Size_{i,t} + \alpha^{\mu} \cdot \mu_{i,t} & \text{if } d_{i,t-1} = 0 \& d_{i,t} = 1 \\ 0 & \text{otherwise} \end{cases}$$

Dynamics - Intertemporal Tradeoffs

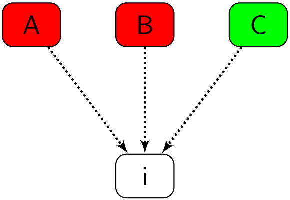
current period t



- $\mu_{i,t} = 2$
- $C_{i,t} = AC_0 + \alpha^{\mu} \cdot 2$
- e.g. package i's belief: $\widehat{p}^1_{A,t+1} = 0$, $\widehat{p}^1_{B,t+1} = 1$

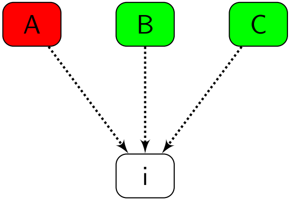
Dynamics - Intertemporal Tradeoffs

current period t



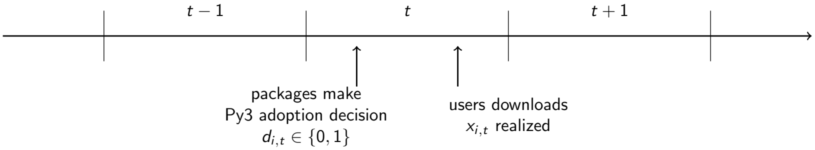
- $\mu_{i,t} = 2$
- $C_{i,t} = AC_0 + \alpha^{\mu} \cdot 2$
- e.g. package i's belief: $\widehat{p}^1_{A,t+1} = 0$, $\widehat{p}^1_{B,t+1} = 1$

forecast t+1



- $\mu_{i,t+1} = 1$
- $C_{i,t} = AC_0 + \alpha^{\mu}$

Model - Timeline



Model

State variables: $S_{i,t} \equiv (x_{i,t-1}, d_{i,t-1}, \nu_{i,t}, \{d_{j,t}, S_{j,t}\}_{j \in U_{i,t}})$

Value function & Bellman equation:

$$V(\mathcal{S}_{i,t}, d_{i,t-1} = 0, v_{i,t}; \theta)$$

$$= \max_{\{d_{i,t+\tau}\}_{\tau=0}^{\infty}} \mathbf{E}_t \{ \sum_{\tau=0}^{\infty} \beta^{\tau} u_{i,t+\tau}(S_{i,t+\tau}, d_{i,t+\tau}) | \mathcal{S}_{i,t}, d_{i,t}; \theta \}$$

$$= \max_{d_{i,t} \in \{0,1\}} u_{i,t}(S_{i,t}, d_{i,t}; \theta) + v_{i,t}^{d_{i,t}} + \beta \mathbf{E}_t V(\mathcal{S}_{i,t+1}, v_{i,t+1} | \mathcal{S}_{i,t}, v_{i,t}, d_{i,t}; \theta)$$

Model

Assuming $v_{i,t}^{d_{i,t}}$ are iid logit errors,

$$
\begin{aligned}
& EV(\mathcal{S}, d=0;\theta) \\
&= \int_{S'} \log\{\sum_{d' \in \{0,1\}} \exp(u(S', d';\theta) + \beta EV(\mathcal{S}', d';\theta))\} d\mathbf{P}_{\mathcal{S}'|\mathcal{S}}
\end{aligned}
$$

$$
\begin{aligned}
\widehat{p}_{i,t}^{1} \equiv & P(d_{i,t}=1|\mathcal{S}_{i,t}, d_{i,t-1}=0, v_{i,t};\theta) \\
= & \frac{exp\{v(\mathcal{S}_{i,t}, v_{i,t}, d_{i,t}=1;\theta)\}}{\sum_{d' \in \{0,1\}} exp\{v(\mathcal{S}_{i,t}, v_{i,t}, d';\theta)\}}
\end{aligned}
$$

MLE: $\theta^* = \arg\max_\theta l(\theta) = \prod_{i=1}^{N} \prod_{t=1}^{T} {\widehat{p}_{i,t}^{0}}^{\mathbf{1}\{d_{i,t}=0\}} {\widehat{p}_{i,t}^{1}}^{\mathbf{1}\{d_{i,t}=1\}}$

Model - Transition Matrix

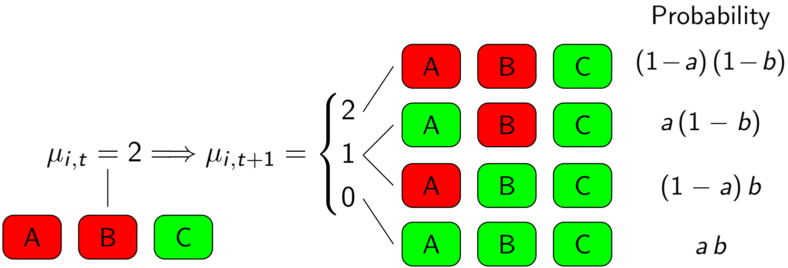
$$
\begin{aligned}
&EV(\mathcal{S}, d = 0; \theta) \\
&= \int_{S'} \log\{ \sum_{d' \in \{0,1\}} \exp(u(S', d'; \theta) + \beta EV(\mathcal{S}', d'; \theta))\} d\underline{\mathbf{P}_{\mathcal{S}'|\mathcal{S}}}
\end{aligned}
$$

State Variables: $S_{i,t} \equiv (x_{i,t-1}, d_{i,t-1}, \nu_{i,t}, \{d_{j,t}, S_{j,t}\}_{j \in U_{i,t}})$
LOM of two important elements: $x_{i,t}$, $\mu_{i,t}$

- $x_{i,t}$: AR1 process
- $\mu_{i,t} \equiv \sum_{j \in U_{i,t}} \mathbb{1}(d_{j,t} = 0)$: the number of Python 3 incompatible dependencies
  - $\mu_{i,t} \equiv \sum_{j \in U_{i,t}} \mathbb{1}(d_{j,t} = 0) \cdot ln(Size_{j,t})$: weighted by size

Model - Transition Matrix - Example

Let $\widehat{p}^{1}_{A,t+1} = a, \quad \widehat{p}^{1}_{B,t+1} = b$



Data - PyPI - Package

name	statsmodels
license	BSD
summary	Estimation and inference for statistical models
author	Josef Perktold, Chad Fulton, Kerby Shedden
version	0.9
requires_dist	numpy
	pandas
	matplotlib
classifiers	Intended Audience :: Science/Research
	Programming Language :: Python :: 2
	Programming Language :: Python :: 3
	Topic :: Scientific/Engineering

Data - PyPI - Package

name	statsmodels
license	BSD
summary	Estimation and inference for statistical models
author	Josef Perktold, Chad Fulton, Kerby Shedden
version	0.9
requires_dist	numpy
	pandas
	matplotlib
classifiers	Intended Audience :: Science/Research
	Programming Language :: Python :: 2
	Programming Language :: Python :: 3
	Topic :: Scientific/Engineering

Data - PyPI - Package

name	statsmodels
license	BSD
summary	Estimation and inference for statistical models
author	Josef Perktold, Chad Fulton, Kerby Shedden
version	0.9
requires_dist	numpy
	pandas
	matplotlib
classifiers	Intended Audience :: Science/Research
	Programming Language :: Python :: 2
	Programming Language :: Python :: 3
	Topic :: Scientific/Engineering

Data - Downloads

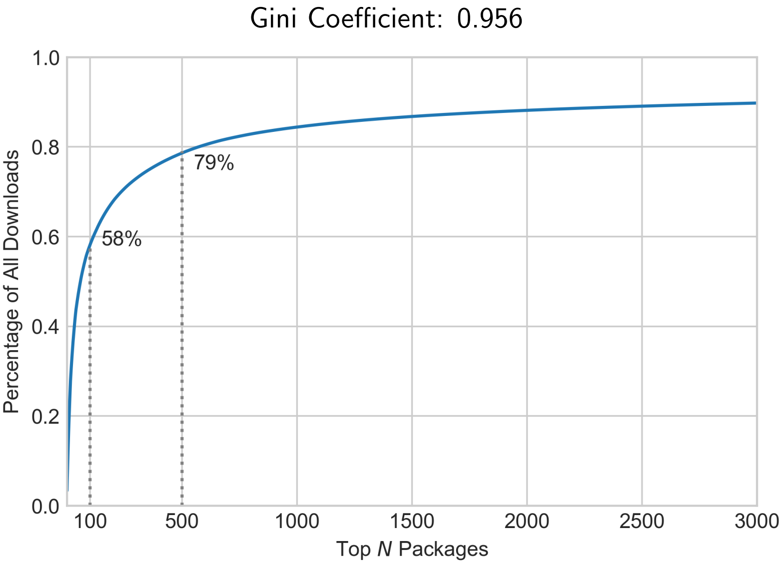
(a) Before 2016: Cumulative Download

upload_time	2014-12-02
python_version	3.4
downloads	41564
filename	statsmodels-0.6.whl
size (bytes)	3969880

(b) After 2016: Individual Download

timestamp	2018-09-01
country_code	FR
filename	statsmodels-0.6.whl
project	statsmodels
version	0.6
python	3.4
system	Mac OS X

Downloads of Top N Packages in 2017



Data Selection

- Total: 140k packages
    - Time Duration (Last Release - First Release Date) $\geq$ 1 Year: 12.9%
    - Downloads Per year $\geq$ 2000: 30.8%
    - Total Number of Releases $\geq$ 5: 38.9%
    - Total Releases / Time Duration $\geq$ 1: 92.4%
    - Some Python 2/3 Support Info Available: 59.9%
    - Initial Support is Python 2 Only: 50.7%
- 4005 packages (3%) and 23267 observations

Introduction

Background

Model

Data

Estimation

Counterfactual

Conclusion

Identification & Estimation

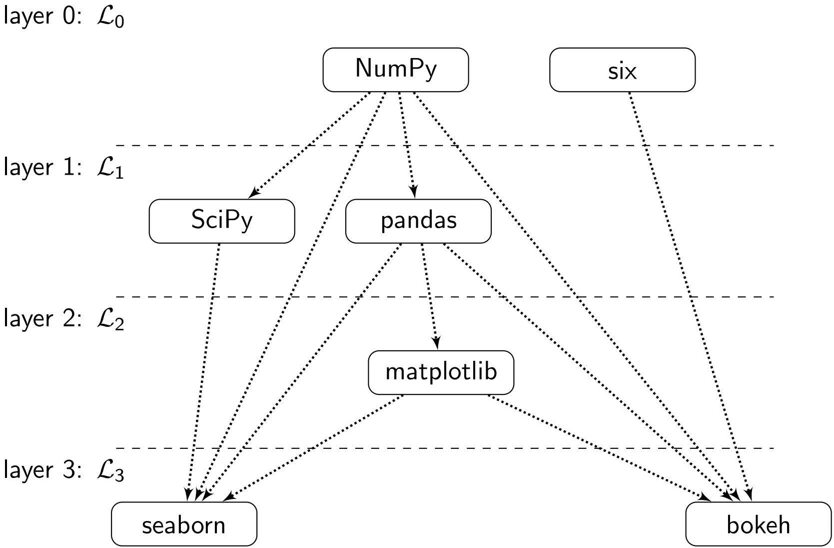
$$\theta = \{\underbrace{\rho_0, \rho_1, \rho_r}_{\theta_D};\ \underbrace{\beta, \alpha^x, AC_0, \alpha^\mu, \alpha^{size}}_{\theta_S}\}.$$

- Step 0: Model Primitive
  - Time Period: Half Year
  - Initial estimate of θ_D^0 from AR1 process:
    $x_{i,t} = \rho_0 + \rho_r \cdot d_{i,t} \cdot r_t + \rho_1 \cdot x_{i,t-1} + \upsilon_{i,t}$
- Step 1: Model Estimation using MLE:

$$\theta_S^1 = \arg\max_{\theta_S} l(\theta_S, \theta_D^0) = \prod_{i=1}^{N}\prod_{t=1}^{T} \widehat{p}_{i,t}^{0\ \mathbf{1}\{d_{i,t}=0\}}\ \widehat{p}_{i,t}^{1\ \mathbf{1}\{d_{i,t}=1\}}$$

- Step 2: Calculate $\widehat{p}_{i,t}^1(\theta_S^1, \theta_D^0)$
  - Update θ_D using $\widehat{p}_{i,t}^1$ as IV for $d_{i,t}$

Estimation - Layered Input-Output Network



Parameter Estimates of User Downloads (θ_D)

	(1) OLS	(2) IV
(ρ_r) $d_{i,t} \times r_t$	0.165***	0.074***
	(0.01)	(0.01)
(ρ_1) $x_{i,t-1}$	0.898***	0.902***
	(0.00)	(0.00)
(ρ_0) Constant	1.069***	1.061***
	(0.02)	(0.02)
N	54230	54230
R^2	0.804	0.803

Parameter Estimates of Adoption Model (θ_S)

Nonlinear	β	0.957***
Parameters (θ_S)		(0.196)
	α^x	0.690***
		(0.053)
	AC_0	-4.743***
		(0.713)
	α^μ	-0.310***
		(0.052)
	α^{size}	-0.219***
		(0.066)
Log Likelihood		-8019
Number of Packages		4005
Number of Observations		23267

- Six-month discount factor $\beta = 0.957$
- Equivalent to monthly $\beta_m = 0.993$

Parameter Estimates of Adoption Model (θ_S)

Nonlinear	β	0.957***
Parameters (θ_S)		(0.196)
	α^x	0.690***
		(0.053)
	AC_0	-4.743***
		(0.713)
	α^μ	-0.310***
		(0.052)
	α^{size}	-0.219***
		(0.066)
Log Likelihood		-8019
Number of Packages		4005
Number of Observations		23267

$$C_{i,t} = AC_0 + \alpha^{size} \cdot Size_{i,t} + \alpha^\mu \cdot \mu_{i,t}$$

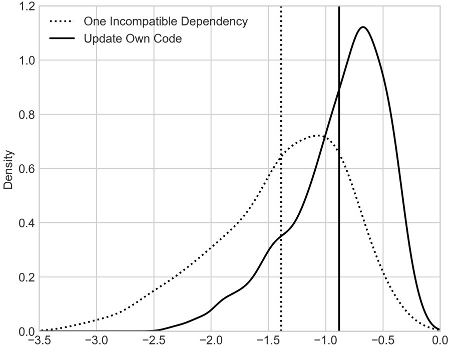
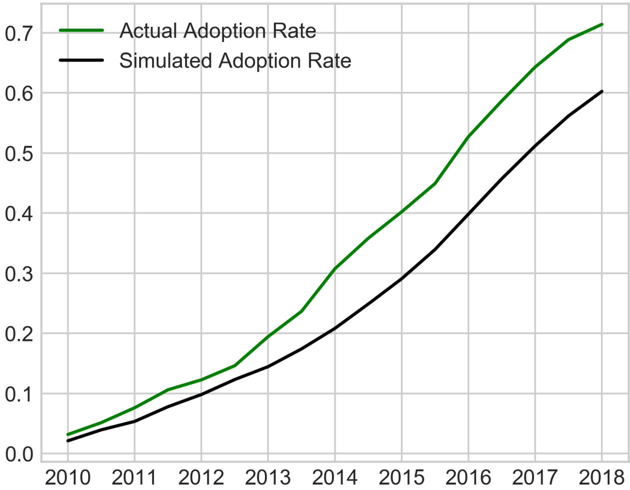


Figure: Variable cost due to one incompatible dependency versus one's own code (convert $\alpha^\mu, \alpha^{size}$ to the same scale)

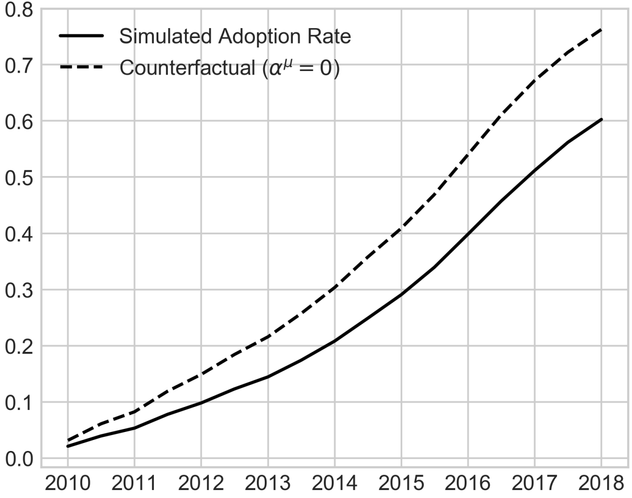
Parameter Estimates with Unobserved Heterogeneity

Nonlinear	β	0.936***	
Parameters (θ_S)		(0.027)	
	α^x	0.854***	
		(0.221)	with probability
	AC_0 (type 1)	-3.513***	76.9%
		(0.447)	
	AC_0 (type 2)	-7.824***	23.1%
		(1.286)	
	α^μ	-0.328***	
		(0.025)	
	α^{size}	-0.114***	
		(0.042)	
Log Likelihood		-8010	
Number of Packages		4005	
Number of Observations		23267	

Model Fit: Actual vs. Simulated Adoption Rates



Model Fit: Actual vs. Simulated Adoption Rates

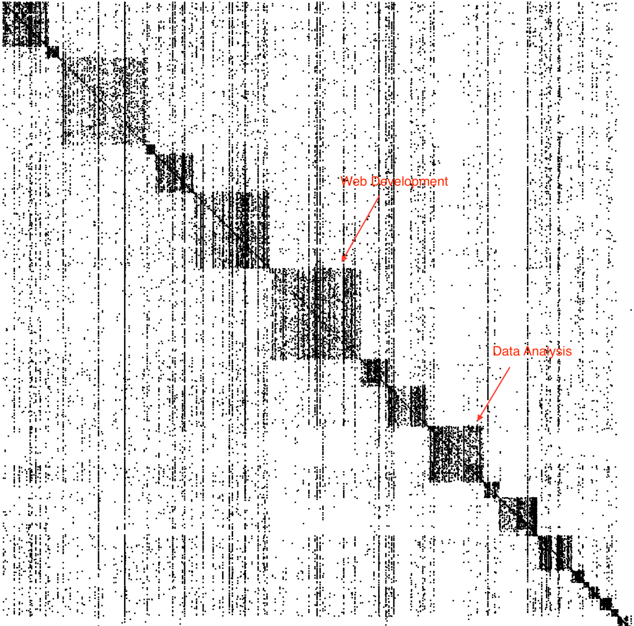


Counterfactual

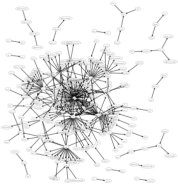
Sponsorship

- Katz and Shapiro (1985) prediction: a new technology is more likely to succeed and spread faster with sponsorship (i.e. someone willing to promote it)
- Python Software Foundation (PSF)

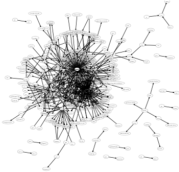
Python Communities



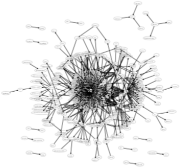
Python Communities



(1) text processing & database

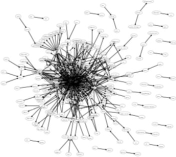


(2) software development & security



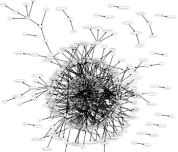
(3) website navigation & processing

(4) software packing



(5) data analysis

(6) cryptography & security



Counterfactual - Community-level Promotion

- Assume that the promotion can increase the expectation of future adoption rate by 10%
  - $x_{i,t} = \rho_0 + \rho_r \cdot d_{i,t} \cdot r_t + \rho_1 \cdot x_{i,t-1} + \upsilon_{i,t}$
  - $\rho_r = 0.097$
  - i.e. $x_{i,t}$ increases by about 0.01 per period, or 1% increase in user downloads every period
- What's the effect on Python 3 adoption rate for the targeted community and other communities?

Changes in Adoption Rate in 2017 with Community Promotion

	1	2	3	4	5	6	7	8	9
1	6.18%	1.13%	-0.47%	-0.40%	0.95%	0.21%	1.08%	0.67%	0.62%
2	0.13%	6.87%	0.88%	0.09%	0.49%	0.05%	1.03%	0.27%	1.29%
3	0.85%	1.23%	7.06%	1.01%	0.77%	0.57%	0.83%	0.11%	-0.19%
4	1.63%	1.12%	1.03%	4.93%	0.90%	0.84%	1.66%	1.24%	1.97%
5	0.85%	1.14%	0.28%	0.05%	7.36%	1.57%	0.84%	2.20%	0.60%
6	1.34%	0.96%	-0.15%	0.64%	2.07%	6.56%	0.97%	-0.74%	-0.86%
7	-0.06%	0.82%	-0.23%	-0.55%	0.71%	0.17%	6.18%	0.91%	0.86%
8	-0.57%	0.12%	-0.35%	-0.32%	1.16%	0.41%	-0.05%	3.86%	0.22%
9	-0.10%	0.14%	1.21%	0.93%	-0.90%	0.93%	1.74%	0.20%	6.70%

Package Characteristics by Community

Community ID	Num. of Packages	Avg Logged Downloads	Avg Age (Years)	Avg Logged Package Size	Avg Num. of Dependencies	Avg Num. of Downstream Packag
1	317	8.442	4.243	3.996	2.389	18.040
2	357	8.687	4.162	3.894	3.393	10.529
3	397	8.150	3.914	3.474	2.605	25.977
4	274	8.702	4.527	3.765	3.623	18.718
5	388	8.154	4.208	5.347	2.785	30.549
6	204	8.786	4.389	4.171	2.827	20.833
7	207	8.280	3.967	3.956	2.727	5.833
8	288	7.877	6.305	4.606	4.751	18.130
9	567	8.326	4.458	3.847	2.407	11.987
10	1006	7.594	3.963	4.050	2.815	6.751

Conclusion

- Dynamic model of technology adoption with input-output networks
- Estimation using a unique dataset of Python packages
- Counterfactual: Community-level promotion

Contribution

- Demonstrate the adverse effects of input-output networks on technology adoption other examples
- Structural model of "network effects"

Future Work / Extensions

- Relax Assumption 2 using Bajari, Benkard and Levin (BBL 2006)
- Other counterfactual policies: individual targeting - what types?
- More flexible unobserved heterogeneity

Thank you!

More questions?

Why Make Python 3 Incompatible with Python 2?

- Painful and conscious decision
- High compatibility cost
- Next research question: What's the optimal compatibility decision?
  - Total compatibility: high development cost for Python, but no adoption cost for users
  - Low compatibility (i.e. almost a new language): low development cost for Python, but high adoption cost for users
  - Python's compatibility decision might not be socially optimal $\longrightarrow$ Payment from users to Python core developers?
  - Variables: objective function, cost curve, quality differences, etc.

back to Py3 Feature

Why Assumption 2

- Computational burden
- AR1 demand process is an approximation of both direct downloads and indirect downloads
- A package has significantly more downloads than downloads of its downstream packages
- Data issue: unclear how much downloads are indirect downloads
- Conditional on downloads level, reduced-form analysis fail to show evidence of coordination

$$|Py3AdoptionDate_i - Py3AdoptionDate_j| = \alpha + \beta \frac{DL_i}{DL_j} + \gamma\, 1(j \in U_i) \frac{DL_i}{DL_j}$$

$$\begin{tikzcd} \boxed{j} \arrow[d, dotted] \\ \boxed{i} \end{tikzcd}$$

	(1) Adoption Date Gap
DL_i/DL_j	-4.801***
	(0.04)
$1(j \in U_i) \times \frac{DL_i}{DL_j}$	-1.214
	(1.97)
Constant	24.907***
	(0.02)
Number of observations	2603384
R2	0.006

Back to model.

Next Version

- Integrate downstream packages' response in $u_{i,t}$ a la Bajari, Benkard and Levin (BBL 2006) commonly used in dynamic games literature
  - Step 1: estimate reduced-form policy function of adoption and include it in $u_{i,t}$
  - Step 2: update the policy function
  - Iteration between Step 1 and 2 until convergence

Back to model.