Success and Geography: Evidence from open-source software

Gábor Békés[1] Julian Hinz[2] Miklós Koren[3] Aaron Lohmann[4]

February 25, 2024

[1]Central European University, KRTK, CEPR
[2]Bielefeld University, Kiel Institute for the World Economy
[3]Central European University, KRTK, CEPR, Cesifo
[4]Bielefeld University, Kiel Institute for the World Economy

Lugano, 2024

Introduction

Big picture questions

Big Picture:

- How dispersed and unorganized developers can create great products.
- How and where good Open Source Software (OSS) is produced.

How and where good Open Source Software (OSS) is produced?

- Are there spatial frictions even though all online?
    - weightless economy – OSS no fixed costs, and no need for face-to-face interaction - pure online.
    - Geography may not significantly impact OSS development.
    - But: spread of talent globally
    - But: spatial frictions in search + matching

How and where good Open Source Software (OSS) is produced?

- Are there spatial frictions even though all online?
  - weightless economy – OSS no fixed costs, and no need for face-to-face interaction - pure online.
  - Geography may not significantly impact OSS development.
  - But: spread of talent globally
  - But: spatial frictions in search + matching

Data:

- Writing code together – Collaboration (Github)
- Using other people's code – imported dependencies (Libraries.io).

Open Source Software (OSS) is everywhere

Open Source Software (OSS) has a vast landscape, `GitHub` hosts over 330 million repositories.

OSS plays an important roles in

- Websites (JavaScript)
- Operating systems (Linux, Android)
- Data (R Tidyverse, Python Pandas, Julia)
- Machine Learning and AI (PyTorch, LLaMA)

OSS mostly free, but present in fee-based platforms

- Overleaf

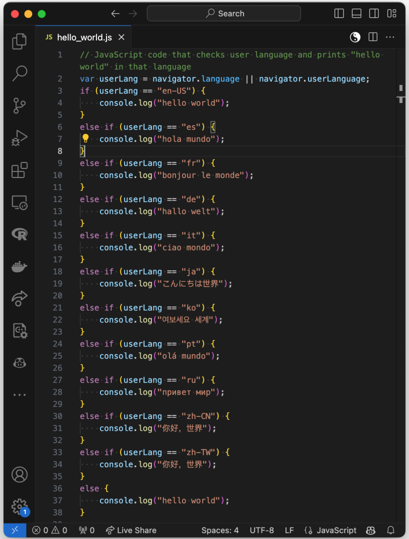
Focus on JavaScript

- `JavaScript` is one of the biggest programming languages

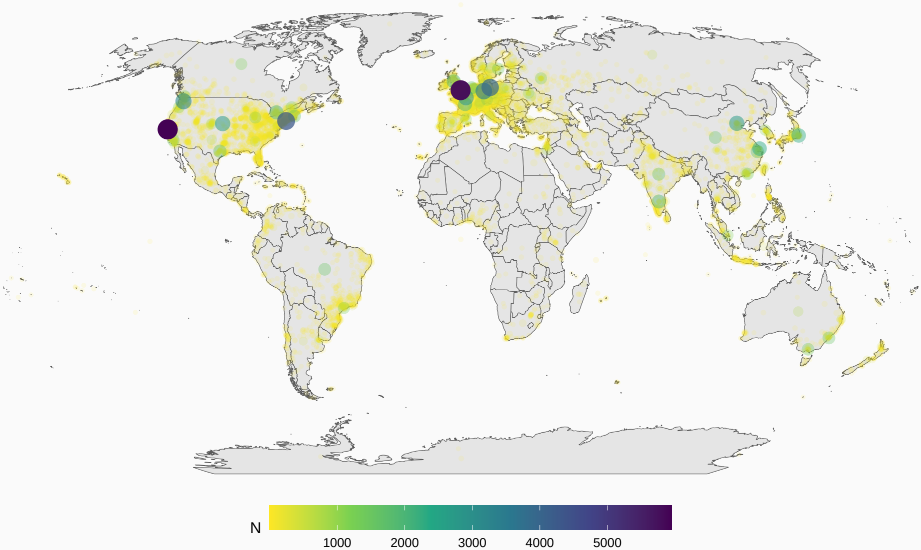
→ used in web development and app development

- `NPM` is a package manager

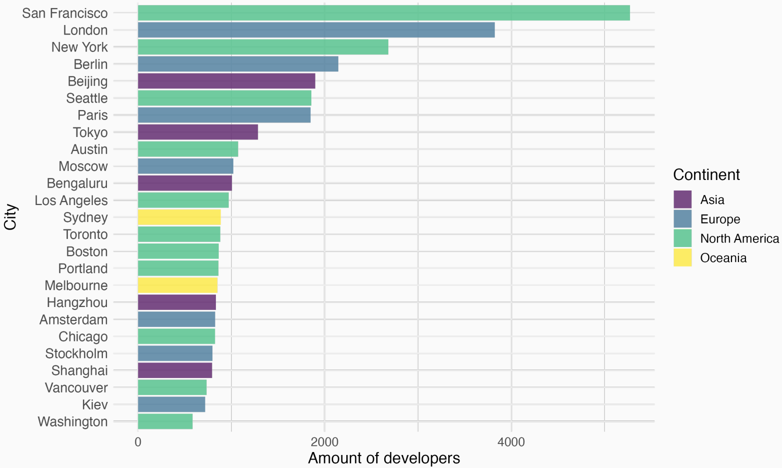
→ organizes packages and provides access



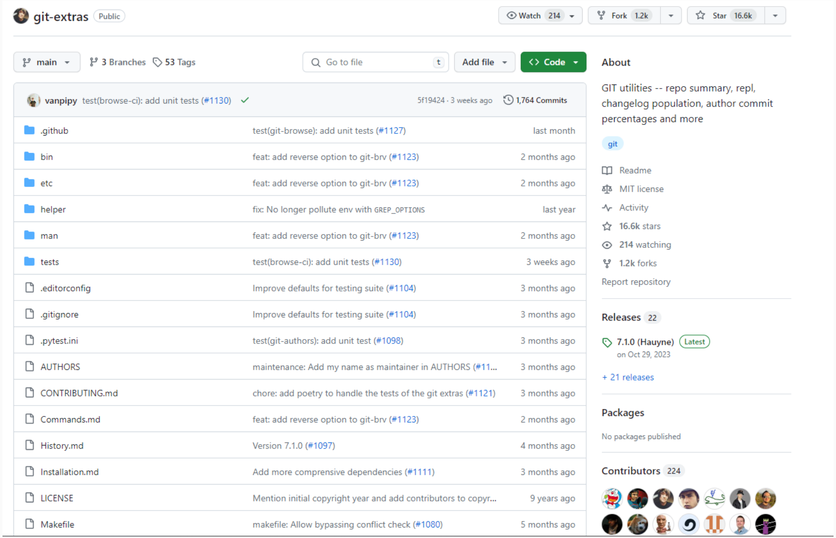
Global industry: Number of JavaScript developer per city



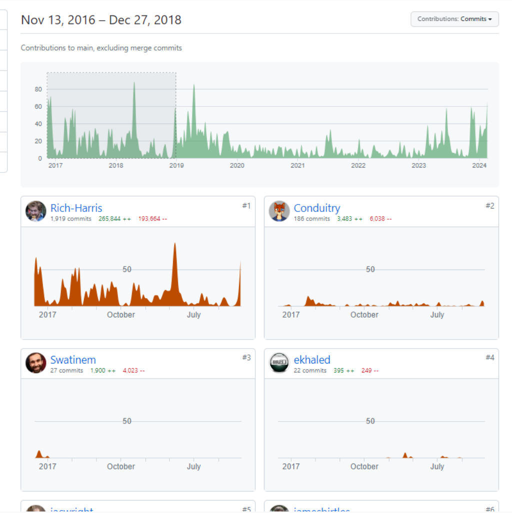
Dispersion and concentration: top cities per number of developers



Collaboration is done mostly online



Collaboration is done mostly online

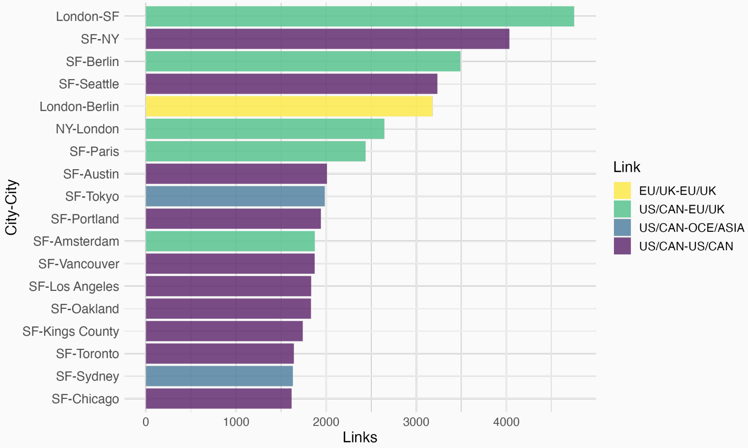


… but personal contacts still matter

- Personal meeting, esp. workplace (CEU, Oracle)
- Local community events, science parks (Xaccelerator)
- Regional events (R Ladies Auckland, VDSG Meetup, PyData Berlin)
- Conferences 1: dozens of events every month such CityJS Berlin, React Summit US,
- Conferences 2: developers directly such as Node-js fwdays23 in Kyiv, where new packages are presented.

- Learn about packages, devs: online forums, Stack Overflow, Twitter

Collaboration across cities is mostly North-North



Most frequent city-pairs for repos developed from 2 cities

Related literature

- **Geographical Distance / Network formation / Agglomeration**: Chaney (2014) Bernard et al. (2019) Davis and Dingel (2019) Bailey et al. (2021)
- **Gravity: Digital:** Blum and Goldfarb (2006) Anderson et al. (2018)
- **Frictions in services:** Stein and Daude (2007) Bahar (2020)
- **Patents and science**: Bircan et al. (2021), Head et al. (2019), Jaffe et al. (1993), Singh (2008) AlShebli et al. (2018)
- **OSS**: Lerner and Tirole (2002) , Laurentsyeva (2019) Wachs et al. (2022) Fackler et al. (2023)

Open source software vs patents and academia

- R&D and patenting
  - Need machines, secrecy, often top-down
  - Distance matters in collaboration
  - More cited patents – geographically focused authors
- Science (math, academic papers)
  - Similar, but often longer projects, not open, F2F important to think and discuss
  - Distance matters in collaboration
  - Major role of top Universities / Centers

Today

- OSS and data
- The role of space in collaboration
  - Gravity
  - Success

Open source software data

Open Source vocabulary

- `Package`: A unit of software, provision of a (bundle of) functionality
- `Project`: A software project offering solution to a use case. Typically one package, but may be more.
- `Repository`: A storage for one project (what we observe)
- `Commit`: The smallest unit of contribution

- `Git`: Distributed version control system for software projects
- `GitHub`: A platform to collaboratively work on software projects

- `Dependency`: An imported package that provides a functionality

Data from GHTorrent and Libraries.io

Collaboration — Working on the same code with others

- `GHTorrent`: Tracks metadata on `GitHub` usage

$\rightarrow$ Commits, locations and user organisations

- Row: One commit from a developer to a repository
- Focus on links: binary if a developer committed at all to a repository

Dependencies — Sourcing of intermediate inputs

- `Libraries.io`: Tracks data on single software repositories

$\rightarrow$ dependency linkages

- Row: An imported dependency (package) to repo 1 from repo 2

$\rightarrow$ Can be mapped to repositories on GitHub

Scope of data

- Data coverage: 2013 – 2019
- We know location as city for developers
- Contributions by 217K developers,
- 300K repos
- 17% of repos have multiple developers (ie have collaboration)
- 70K organizations, with 120K developers

Sample design: exclude later arrival, bug-fixers

We focus on collaborating partners, who are likely to have interaction, joint decisions. Exclude

1. Bugfixers – as external "consultants" who come in help solve a problem
   - Less than 4 commits or 1% of commits — less than 10 commits total
2. Late arrivals – developers who take over maintenance or add important extensions late
   - Developers who first commit 730 days after the first commit

As we look at dynamics, we focus on projects we see the first commit, ie after 2013.

Raw data to regressions

- Collaboration – link developers who contribute to the same repo.
- Dependencies – link developers from one package using another

- One observation is one link
- Aggregated at city (city pair) level

Collaboration

- Start with the developer's link to a repository (via commits)
- Directed but (mostly fully) symmetric
- Transform it to developer to developer links
- Aggregate at city level

Links in the contribution network

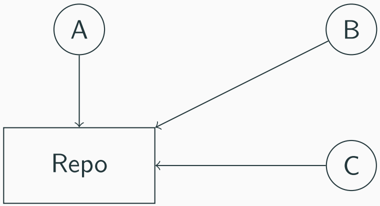


Figure 1: Developers commiting to a repository.

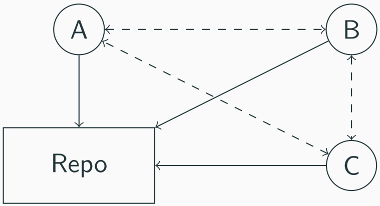


Figure 2: Developers commiting to a repository including implied contributor to contributor links.

Solid lines are what we **observe**. Dashed lines is what we **infer**.

Aggregation – weights

- In a repo, all developers create links with each other
- If two people have 3 repo together, will generate 3 links

- Also look at *intensive* margin – weighted by commits

Organisations

- Github collaboration system
- Mostly amateurs (like CEU Econ)
- Includes corporations (like Oracle)

- Today: mostly focus outside organizations

Estimating gravity

Gravity: finding a partner

- The role of distance in finding a partner
- Search and maintenance

- Each coder can choose any partner: logit
- Aggregate + transform: Poisson at city pair level: number of links as function of distance
- *(Yes, like structural gravity: PPML, FEs)*

MORE: ⏩ From logit to Poisson

Gravity: finding a partner

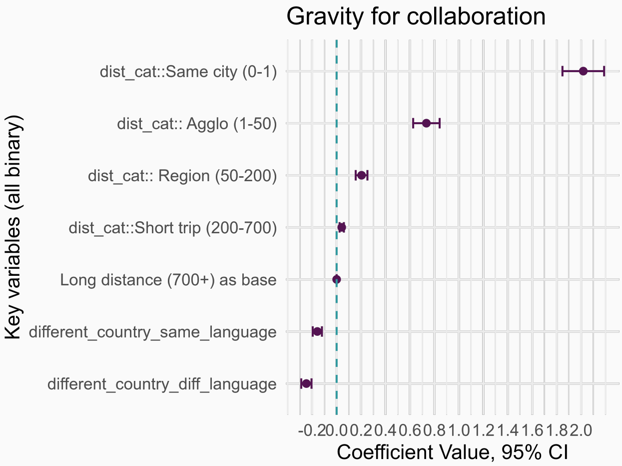
$$\Pr(Y_{od}|x_o, x_d, d_{od}) \approx \text{Poisson}[N_o \times N_d \times \exp(\beta_1 x_o + \beta_2 x_d + \beta_3 d_{od})]$$

- Outcome: Number of links between cities o, d
- d_{od} Distance measured as a set of indicators / log-linear
- Origin and destination city FE
- $N_o \times N_d$ -Exposure: Number of developers in city o x d

Modelling search and maintenance costs

- Meeting – distance in terms of travel
  - Same city – e.g. universities, office parks
  - Agglomeration (1-50km) – regional events
  - Regional (50-200km) – national conferences
  - Short trip (200-700km) – big conferences
  - Beyond 700km (*as base*) – global events
- Travel difficulty
  - Crossing borders
  - Crossing borders — different language

Results 1: More work together when closer



Gravity 1: N of links between cities declines with distance

Dependent Variable:	N of links between contributors		
Model:	(1)	(2)	(3)
ln_dist	-0.1650***		
	(0.0083)		
same_org	5.609***	5.556***	
	(0.0871)	(0.0855)	
dist_cat = Samecity(0-1)		1.746***	2.018***
		(0.0772)	(0.0858)
dist_cat = Agglo(1-50)		0.6351***	0.7344***
		(0.0724)	(0.0873)
dist_cat = Region(50-200)		0.1905***	0.2039***
		(0.0319)	(0.0307)
dist_cat = Shorttrip(200-700)		0.0245*	0.0416***
		(0.0127)	(0.0101)
different_country_same_language		-0.1749***	-0.1581***
		(0.0215)	(0.0184)
different_country_diff_language		-0.2856***	-0.2476***
		(0.0369)	(0.0322)
Pseudo R^2	0.84163	0.84385	0.86084
Observations	3,634,243	3,636,122	3,478,716

Origin, destination city FE, Clustered (city_destination & city_origin) standard-errors in parentheses

Results 2: Commits as kinda intensive margin

- Look at commits – number of changes in code
- Bit like extensive margin

Gravity 2: Co-location = more intensive work

Dependent Variables:	N links	commit share
Model:	(1)	(2)
Variables		
dist_cat = Samecity(0-1)	2.018***	0.7564***
	(0.0858)	(0.1309)
dist_cat = Agglo(1-50)	0.7344***	0.1838
	(0.0873)	(0.1410)
dist_cat = Region(50-200)	0.2039***	0.0906
	(0.0307)	(0.0795)
dist_cat = Shorttrip(200-700)	0.0416***	-0.0192
	(0.0101)	(0.0399)
Fixed-effects		
city_destination	Yes	Yes
city_origin	Yes	Yes
Fit statistics		
Pseudo R^2	0.86084	0.52444
Observations	3,478,716	451,423

Origin, destination city FE, Clustered (city_destination & city_origin) standard-errors in parentheses

Robustness

- Maybe a few very large repositories dominate and flatten the curve. No
- Also no huge difference excluding few largest cities

Estimating success and dispersion

Success (popularity) and spatial dispersion

- Popularity = measures the number of other packages which declare a dependency on a the repository in NPM
- Measures on spatial dispersion
- Controls

Success (popularity) and spatial dispersion

$$\Pr(Y_i|.) \approx \text{Poisson}[\exp(\beta_1 cities_i + \beta_2 countries_i) + \gamma \mathbf{Z}]$$

- Outcome: Number of repos importing this repo i
- $countries_i$ number of countries
- $cities_i$ number of cities
- **Z: f(number of developers), f(age of project) + other controls**

Methods: models

Mechanic controls

- Number of developers (as categorical)
- Age of project, ys (as categorical)

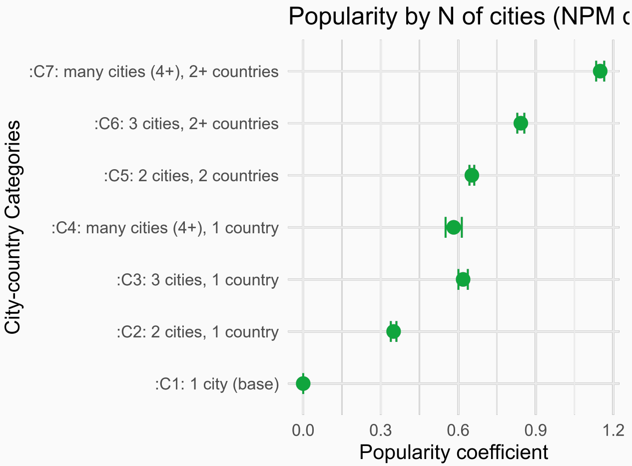
Coder quality

- Size of the city of coder(s) – top 3 coders
- Quality of coder(s) as measured by number of stars– top 3 coders

Commits

- sum commits (squared)

Results 1: More popular dependency - higher spatial dispersion



Results 1: More popular dependency - higher spatial dispersion

Dependent Variable:	N Dependents (NPM)		
Model:	(1)	(2)	(3)
Constant	1.745***	1.148***	1.673***
	(0.0548)	(0.0857)	(0.0674)
Count of cities	0.4075***	0.2306***	
	(0.0403)	(0.0491)	
Count of countries	0.3431***	0.3057***	
	(0.0628)	(0.0637)	
City cat × CI2 × 2cities			0.3851***
			(0.0813)
City cat × CI3 × 3cities			0.4925***
			(0.1242)
City cat × CI4 × manycities(4+)			0.6543***
			(0.1642)
Country cat × CO2 × 2countries			0.2461***
			(0.0813)
Country cat × CO3 × manycountries(3+)			0.6269***
			(0.1462)
Age, N_Dev	No	Yes	Yes
Commits	No	No	No
Coders	No	No	No
Pseudo R^2	0.05100	0.11532	0.11586
Observations	36,491	36,491	36,491

What's behind this correlation?

- Packages written by *more* dispersed people will be used more.
    - Is it because a diverse group reaches diverse regions generating use? (reverse causality-ish)
        - Let us look at other packages using ours as dependency

What's behind this correlation?

- Packages written by *more* dispersed people will be used more.
  - Is it because a diverse group reaches diverse regions generating use? (reverse causality-ish)
    - Let us look at other packages using ours as dependency
  - Is this generated by a sorting model?
    - Think about source of sorting
    - Condition on selection

Preparation: Aggregating dependencies to city level

- We observe a repository importing another one as dependency.
- Directed, not symmetric
- Transform it to developer to developer links
    - Use knowledge of producers of the dependency as well
- Aggregate at city level

Links in the dependency network

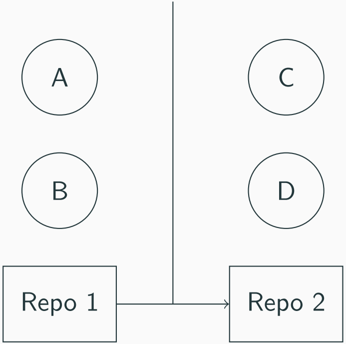


Figure 3: Dependency of repository 1 on repository 2 with the respective developers.

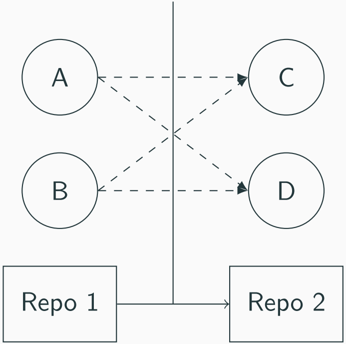


Figure 4: Dependency of repository 1 on repository 2 with the respective developers. Dashed lines indicate implied links between developers.

Again, solid lines are what we **observe**. Dashed lines is what we **infer**.

1. Not reverse causality - dependency use just mildly spatial

Dependent Variables:	contr_n_links	dep_value
Model:	(1)	(2)
Variables		
dist_cat = Samecity(0-1)	2.018***	0.0754***
	(0.0858)	(0.0138)
dist_cat = Agglo(1-50)	0.7344***	0.0805***
	(0.0873)	(0.0127)
dist_cat = Region(50-200)	0.2039***	0.0254***
	(0.0307)	(0.0095)
dist_cat = Shorttrip(200-700)	0.0416***	0.0045
	(0.0101)	(0.0036)
different_country_same_language	-0.1581***	-0.0222***
	(0.0184)	(0.0082)
different_country_diff_language	-0.2476***	-0.0499***
	(0.0322)	(0.0115)
Pseudo R^2	0.86084	0.98866
Observations	3,478,716	3,202,202

Origin, destination city FE, Clustered (city_destination & city_origin) standard-errors in parentheses

2. Models of sorting

- Coders vary by skill, match within and across cities randomly, large cities have better coders. Opposite correlation

- Coders vary by many types of skill. FC of matching across cities. Better coders self-select into diverse teams producing better code. Selection on coder skills.

MORE: More on a sketch of a theory

Selection?

- Coder quality: Number of repos, number of followers
- Location: Size of the city (N of total coders)

- Commits: number of commits to repo

Results 2: Selection? Partialing out coder quality and commits

Dependent Variables:	N Dependents (NPM)			N commits
Model:	(1)	(2)	(3)	(4)
Count of cities	0.2306***	0.1906***	0.2818***	-0.1087***
	(0.0491)	(0.0488)	(0.0509)	(0.0358)
Count of countries	0.3057***	0.3243***	0.2904***	0.0355
	(0.0637)	(0.0634)	(0.0637)	(0.0583)
Sum of commits (ln)			0.3330***	
			(0.0202)	
Constant	1.148***	-0.0641	-1.746***	5.090***
	(0.0857)	(0.1140)	(0.1507)	(0.0854)
Age, N_Dev	Yes	Yes	Yes	Yes
Coders	No	Yes	Yes	Yes
Commits	No	No	Yes	–
Pseudo R^2	0.11532	0.15056	0.18345	0.26074
Observations	36,491	36,491	36,491	36,491

Success and dispersion

- Compare coders of similar quality based in similar locations
- Exclude success driven by bigger spatial reach of developers

- Group of diverse coders will create more successful projects
- Even conditioning on intensity of collaboration (commits)

Data issues

- Organization
  - Results robust, but something is going on in orgs.
- Missing city info
  - Half of city information is missing – could be non-random
  - For many we know the organisation they work for – l8r
  - Results robust to filter on 2-3 member teams with known location

Discussion

Summary

- Location matters even for coding

Summary

- Location matters even for coding
- Will the best coders congregate in big cities to create best code?

Summary

- Location matters even for coding
- Will the best coders congregate in big cities to create best code?
- No. Spatially dispersed developers create code that is more widely adopted.

- Sorting matters: good coders write good code used by more. But not explains
- It's also no more effort – when looking at commits, we see the opposite...

Summary

- Location matters even for coding
- Will the best coders congregate in big cities to create best code?
- No. Spatially dispersed developers create code that is more widely adopted.

- Sorting matters: good coders write good code used by more. But not explains
- It's also no more effort – when looking at commits, we see the opposite...

- There is something else. Maybe higher fixed costs of starting a diverse project generates a selection on project ideas.

References

AlShebli, Bedoor K., Talal Rahwan, and Wei Lee Woon, “The preeminence of ethnic diversity in scientific collaboration,” *Nature communications*, 2018, *9*, 1–10.

Anderson, James E, Ingo Borchert, Aaditya Mattoo, and Yoto V Yotov, “Dark costs, missing data: Shedding some light on services trade,” *European Economic Review*, 2018, *105*, 193–214.

Bahar, Dany, “The hardships of long distance relationships: time zone proximity and the location of MNC’s knowledge-intensive activities,” *Journal of International Economics*, 2020, *125*, 103311.

Bailey, Mike, Abhinav Gupta, Sebastian Hillenbrand, Theresa Kuchler, Robert Richmond, and Johannes Stroebel, "International Trade and Social Connectedness," *Journal of International Economics*, Mar 2021, *129*, 103418.

Bernard, Andrew B, Andreas Moxnes, and Yukiko U Saito, "Production networks, geography, and firm performance," *Journal of Political Economy*, 2019, *127* (2), 639–688.

Bircan, Cagatay, Beata Javorcik, and Stefan Pauly, "Creation and Diffusion of Knowledge in the Multinational Firm," 2021. Working Paper.

Blum, Bernardo S and Avi Goldfarb, "Does the internet defy the law of gravity?," *Journal of international economics*, 2006, *70* (2), 384–405.

Chaney, Thomas, "The network structure of international trade," *The American Economic Review*, 2014, *104* (11), 3600–3634.

Davis, Donald R and Jonathan I Dingel, "A spatial knowledge economy," *American Economic Review*, 2019, *109* (1), 153–170.

Fackler, Thomas, Michael Hofmann, and Nadzeya Laurentsyeva, "Defying Gravity: What Drives Productivity in Remote Teams?," Technical Report, LMU CRCT Discussuion Paper 427 2023.

Head, Keith, Yao Amber Li, and Asier Minondo, "Geography, Ties, and Knowledge Flows: Evidence from Citations in Mathematics," *The Review of Economics and Statistics*, 10 2019, *101* (4), 713–727.

Jaffe, Adam B, Manuel Trajtenberg, and Rebecca Henderson, "Geographic localization of knowledge spillovers as evidenced by patent citations," *Quarterly journal of Economics*, 1993, *108* (3), 577–598.

Laurentsyeva, Nadzeya, "From friends to foes: National identity and collaboration in diverse teams," Technical Report, CESifo Discussion Paper 2019.

Lerner, Josh and Jean Tirole, "Some simple economics of open source," *The journal of industrial economics*, 2002, *50* (2), 197–234.

Stein, Ernesto and Christian Daude, "Longitude matters: Time zones and the location of foreign direct investment," *Journal of International Economics*, 2007, *71* (1), 96–112.

Wachs, Johannes, Mariusz Nitecki, William Schueller, and Axel Polleres, "The Geography of Open Source Software: Evidence from GitHub," *Technological Forecasting and Social Change*, 2022, *176*, 121478.

Behind Poisson 1: Individual matching decision

Collaboration or dependency link between developer i and j,

$$\Pr(Y_{ij} = 1 | x_i, x_j, d_{ij}) = \Pi(\beta_1 x_i + \beta_2 x_j + \beta_3 d_{ij})$$

with

$$\Pi(z) = e^z / (1 + e^z)$$

the logistic function

Assumption: Independence across links, add fixed effects

Behind Poisson 2: Aggregate to Poisson

In practice, distance only varies at the city level. Take origin city o and destination city d.

$$Y_{od} := \sum_{i \in o} \sum_{j \in d} Y_{ij}$$

$$\Pr(Y_{od} | x_o, x_d, d_{od}) = \text{Binomial}[N_o \times N_d, \Pi(\beta_1 x_i + \beta_2 x_j + \beta_3 d_{ij})]$$

Here $N_o \times N_d$ is the total number of *potential* links between cities o and d.

When Π is small, we aggregate i into cities o, and j into cities d

$$\Pr(Y_{od} | x_o, x_d, d_{od}) \approx \text{Poisson}[N_o \times N_d \times \exp(\beta_1 x_o + \beta_2 x_d + \beta_3 d_{od})]$$

Behind Poisson 3: Having exposure is key

We may also look at a subsample (like users not in the same GitHub organization)

$$Y_{od,\text{not org}} := \sum_{i \in o} \sum_{j \in d, j \notin \text{org}(i)} Y_{ij}$$

This changes the *exposure variable*,

$$\Pr(Y_{od,\text{not org}} | x_o, x_d, d_{od}) \approx \text{Poisson}[N_{od,\text{not org}} \times \exp(\beta_1 x_o + \beta_2 x_d + \beta_3 d_{od})],$$

with $N_{od,\text{not org}}$ the number of user pairs in city o, d, *not sharing* an organization.

Important: $N_{od,\text{not org}}$ may be zero.

What is a Poisson regression?

First-order conditions for Maximum Likelihood:

$$\sum_o \sum_d x_i [Y_{od} - N_{od} \exp(\beta_1 x_o + \beta_2 x_d + \beta_3 d_{od})] = 0$$
$$\sum_o \sum_d x_j [Y_{od} - N_{od} \exp(\beta_1 x_o + \beta_2 x_d + \beta_3 d_{od})] = 0$$
$$\sum_o \sum_d d_{ij} [Y_{od} - N_{od} \exp(\beta_1 x_o + \beta_2 x_d + \beta_3 d_{od})] = 0$$

- Level (not log) error terms are orthogonal to RHS variables.
- Exposure variable has fixed exponent of 1 ($\approx$ weighting).
- Standard errors computed from GMM, not ML. E.g., we allow for two-way city clustering.

What is an observation?

Two interpretations:

1. 10 billion potential developer pairs
2. 3.7 million city pairs

Model sketch

- Production of code is driven by utility gains of creating code used by many people
- Coders are heterogeneous in coding quality.
- Coders collaborate with others when
  - Task is too complex for a single person. Economies of scale.
  - ...
- There is selection into projects: best coders write most complex packages.

Model sketch 2: The role of geography

- Coders are dispersed geographically – located in a discrete set of N_c cities
  - City size (number of coders) Pareto distributed
  - Size may be driven by first geography (later), such as proximity to University, tech firms or the beach.

- Heterogeneity of coders: at every location, their distribution is Pareto

- Random matching: simple random selection of collaborators
- Assortative matching: Coders match with coders of same quality

Model: self selection of coders

- If best programmers are in big cities (Pareto with different k across cities): size and quality correlated
- Top coders coming from large cities will produce best code –¿ more popular code.
- Best code will come more than proportionally from large cities

- Assortative matching reinforces this aspect, as big city coders will only work with big city coders

- Best code written by people in top cities (like SF) – homogeneity

Model: There are search costs

- Costs of setting up a partnership and maintaining it
- Search costs of inputs (code chunks)
  - Written together – finding a collaborator
  - Using already published code – finding a package
- Search costs vary with distance – lower inside the city

Model: Coder heterogeneity

- There is a set of possible coding skills, S
- Coders randomly vary in each skill, $s = 1, 2, 3 ... S$
- Two coders who are on average same quality still have difference and can benefitb from collaboration, where the pair's skill is max

Model: Dispersion forces

- Coders differ to some extent, and so search is needed
- There is a search cost, higher for other cities
- Better coders pay higher search cost and hence can search a larger pool across cities

Model: Additional aspects

- Face to face matters when creating complex projects.
- Some cities specialize in some tasks