When dispersed teams are more successful: Theory and evidence from software

Gábor Békés, Julian Hinz, Miklós Koren, and Aaron Lohmann

CEU, KRTK, CEPR, University Bielefeld and IfW Kiel

2025-05-09

Research questions

1. How and where is open source software developed?
2. Do spatially dispersed developers produce quality software?
3. How do team formation and collaboration affect the success of OSS projects?

Economics of open source

- Incentives to produce and contribute to open source
- Spatial dispersion and its interactions with quality

Why Open Source Software (OSS)?

- Software is everywhere and, more specifically, OSS is everywhere
  - 98% of commercial software uses OSS (Synopsys 2023).
  - OSS is powering Machine Learning, AI development and embedded systems.
- OSS is huge
  - Hoffmann, Nagle, and Zhou (2024) estimate demand side as 8.8 trilion USD; GitHub nowadays has over 100 million developers
- OSS is observable
  - Due to the `git` paradigm almost everything is recorded

What we see in the data: The example of `ggplot2`

Figure 1: Hadley Wickham

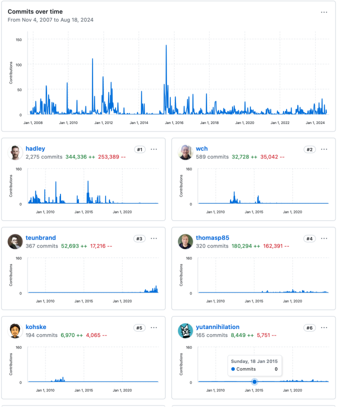


Figure 2: Commits in ggplot2

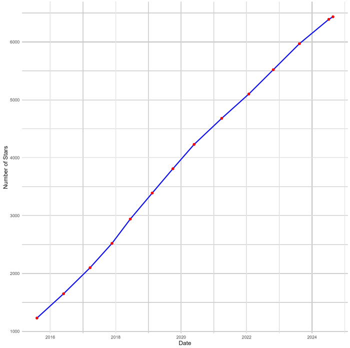
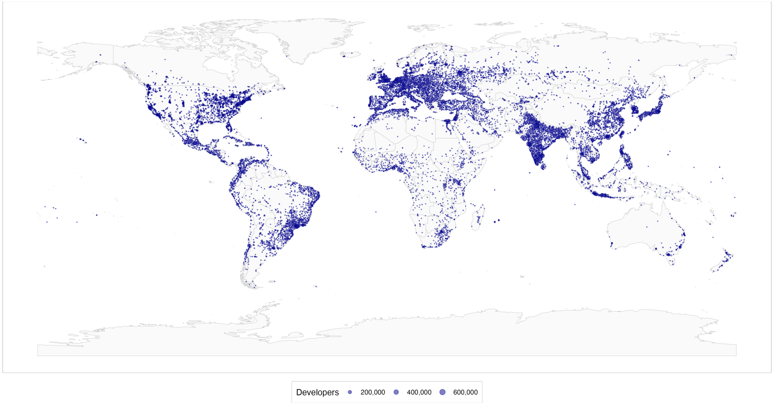


Figure 3: ggplot2 stars over time

A truly global industry



Team formation follows gravity

Dependent Variable:	Collaboration count	
Model:	(1)	(2)
Variables		
Same Country	1.837***	0.9497***
	(0.0522)	(0.0878)
Log Distance	-0.8720***	-0.8634***
	(0.0118)	(0.0119)
Common Official Language		1.000***
		(0.0764)
Common Religion		0.8163***
		(0.1077)
Fixed-effects		
Developer 1 City	Yes	Yes
Developer 2 City	Yes	Yes
Fit statistics		
Observations	21,687,649	21,242,884
Squared Correlation	0.06270	0.06520

Literature

- **Production in teams:** Jarosch, Oberfield, and Rossi-Hansberg (2021) ; Herkenhoff et al. (2024) ; Freund (2022) ; Kerr and Kerr (2018)
  Our contribution: A model for global team formation which has selection as a main mechanism.
- **Gravity/International Trade:** Eaton and Kortum (2002) ; Atkin, Chen, and Popov (2022) ; Head, Li, and Minondo (2019)
  Our contribution: Gravity estimates for team formation in OSS.
- **OSS**: Lerner and Tirole (2002) ; Fackler and Laurentsyeva (2020) ; Wachs et al. (2022) ; Betty et al (2025)
  Our contribution: Providing more descriptive statistics, making use of data and combining several data sources.

This paper

Model global *team formation*, *collaboration*, and *performance* for open-source software projects.

Test model predictions using data from GitHub 2018-2024.

(later) Structurally estimate the model. Conduct counterfactuals about digital fragmentation, AI, and the Global South.

Team formation, collaboration, and performance

Timing

1. Developers have an idea (uncertain quality)
2. Two developers meet at random (incomplete information)
3. Decide whether to do a project together
4. Software quality depends on ideas and effort
5. Users download it, distributing kudos to developers

Production function

Software quality depends on *design* z and *implementation* e:

$$y_p = z_p^{\nu} e_p^{1-\nu}$$

Diminishing returns to implementation effort for a given design $\rightarrow$ rents to good design.

Incentives

Developers care about software quality (receive "kudos") and incur *convex* costs of effort.

Benchmark 1: Solo project

A solo developer maximizes rent from design:

$$\max_e z_p^\nu e^{1-\nu} - c_{ip}\frac{e^{1+\gamma}}{1+\gamma}$$

with

$$e_p^* = (1-\nu)^{1/(\gamma+\nu)} z_p^{\nu/(\gamma+\nu)} c_{ip}^{-1/(\gamma+\nu)}$$

and

$$y_p^* = (1-\nu)^{(1-\nu)/(\gamma+\nu)} z_p^{\nu(1+\gamma)/(\gamma+\nu)} c_{ip}^{-(1-\nu)/(\gamma+\nu)}.$$

Effort and project quality increase in design quality and decrease in development cost.

Rent is $(\gamma+\nu)/(1+\gamma)$ share of software project quality.

Benchmark 2: Market-based collaboration

Hire another developer, paying her effort costs, to maximize

$$z_p^{\nu}(e_{1p} + e_{2p})^{1-\nu} - c_{2p}\frac{e_{2p}^{1+\gamma}}{1+\gamma}$$

Working in a team may be better. Developer 1 can increase the rents to her design.

Good model for in-house, proprietary software.

Hard in OSS, where no money is involved and effort is not contractible.

Benefits of collaboration

1. Defraying effort costs: getting tired, bored, diminishing return to specialized expertise.
2. Complementarity of skills (not modeled): developers specialize in different parts of the code.
3. Knowledge sharing: design quality may improve.

Incentivizing team members

Contracting, monetary transfer assumed impossible.

Kudos is valued, but not equally by all members (attribution problem, job market opportunities).

Team members can split kudos in *ex ante* agreed $(1-\phi):\phi$ share.

Kudos sharing and participation constraints

$$
\begin{aligned}
\max_{e_1} (1-\phi) z_p^{\nu} (e_1 + e_2)^{1-\nu} - c_1 \frac{e_1^{1+\gamma}}{1+\gamma} \quad &\geq \kappa_{1p} \\
\max_{e_2} \phi z_p^{\nu} (e_1 + e_2)^{1-\nu} - c_2 \frac{e_2^{1+\gamma}}{1+\gamma} \quad &\geq \kappa_{2p}
\end{aligned}
$$

with both developers earning more than the outside option κ_{ip}.

Frictions in collaboration

Communication

Hard to work on someone else's code without frequent communication (may depend on distance):

$$c_{2p} := \tau_{12} c_2$$

Kudos value

Distant developers may enjoy less of the kudos (e.g., cannot turn kudos into job opportunities, may depend on distance)

$$\kappa_{2p} := d_{12} \kappa_2$$

Information structure

Developers cannot credibly signal their skill / quality of their idea.

At the time of team formation, project quality is uncertain.

Equilibrium team formation

Given kudos sharing rules and frictions, team formation satisfies participation constraints

$$\begin{aligned} \mathsf{E}_1[(1-\phi)y_p^*] &\geq y_{1p}^* \\ \mathsf{E}_2(\phi y_p^*) &\geq d_{12}\tilde{\kappa}_2 \end{aligned}$$

with $\tilde{\kappa}_2 = \kappa_2(1+\gamma)/(\gamma+\nu)$.

Distance frictions to team formation

Far-away developers less likely to join project:

- less expected rent due to miscommunication / lower effort
- higher outside option due to rent appropriation frictions

Selection of projects by quality

Only better projects get produced.

Selection of developers by distance

Remote developers (higher d_{12}) have to expect

- higher project quality or
- higher kudos share to join.

Kudos sharing rule: Credit where credit due

$$z_p := \max\{z_1, z_2\}$$

Developer with the winning design gets higher share:

$$\phi := \begin{cases} \phi_H & \text{if } z_2 \geq z_1 \\ \phi_L & \text{if } z_2 < z_1 \end{cases}$$

with $\phi_H > \frac{1}{2} > \phi_L$

Good developers have *disproportionately* higher expected payoff from collaborative projects $\rightarrow$ positive assortative matching.

Testable predictions

Testable predictions

Team formation

1. More skilled developers more likely to join collaborative projects.
2. Distant developers less likely to join collaborative projects.
3. Distant developers on collaborative projects have higher skill.
4. (if $\phi_H \gg \phi_L$) Positive assortative matching of developers by skill.

Collaboration

5. Collaborative projects more successful than solo projects.
6. Lead developer works more on project than follower.
7. Skilled developers work more.
8. Followers work more on project with a skilled leader.
9. Distant followers work less.

Project success

10. Skill and effort both matter for project success.
11. Projects with distant developers are more successful.

Measurement

Lead developer

- Person starting the project
- If developers join in same month, person with most commits in the first month

Skill

- Experience: Number of commits before project start

Quality

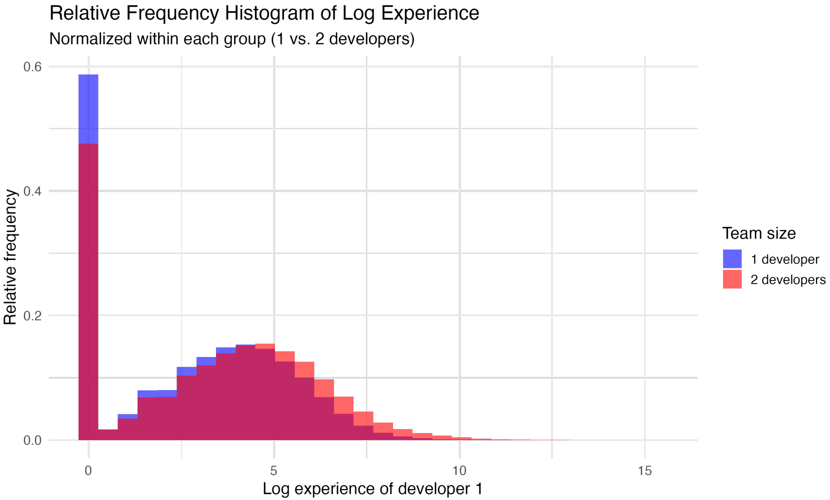
- Number of stars after 6 months
- (Number of forks, number of downloads)

Effort

- Number of commits in first 6 months

Selection into collaboration

More skilled developers are overrepresented in collaborative projects.

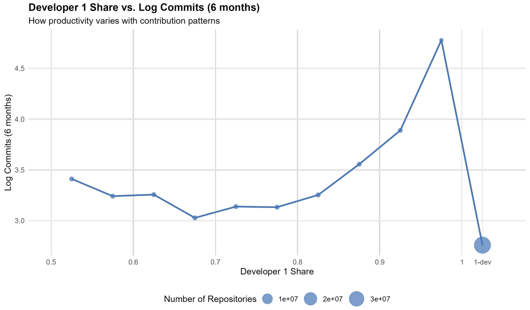


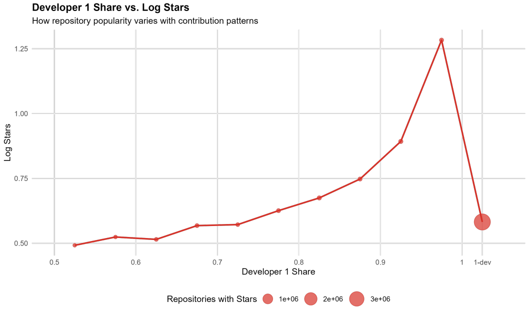
More experienced developers are joined by more experienced developers

Dependent Variable:	Follower experience				
Model:	(1)	(2)	(3)	(4)	(5)
Variables					
Lead developer experience (log)	0.1079***	0.0896***	0.0897***	0.0809***	0.3108***
	(0.0001)	(0.0009)	(0.0009)	(0.0009)	(0.0017)
Total commits (log)			0.1143***	0.1300***	0.1326***
			(0.0025)	(0.0024)	(0.0046)
Lead developer commits (log)			-0.1000***	-0.1142***	-0.1275***
			(0.0024)	(0.0023)	(0.0044)
Distance (log)				0.0362***	-0.0042***
				(0.0004)	(0.0007)
Fixed-effects					
language_time_fe		Yes	Yes	Yes	Yes
Fit statistics					
Observations	2,340,003	2,339,777	2,339,777	2,338,828	862,462
Squared Correlation	0.11041	0.14524	0.14695	0.15897	0.28787
Pseudo R^2	0.04672	0.06355	0.06411	0.06988	0.17988
BIC	11,109,367.1	11,008,745.1	11,002,287.6	10,930,464.8	3,009,542.6

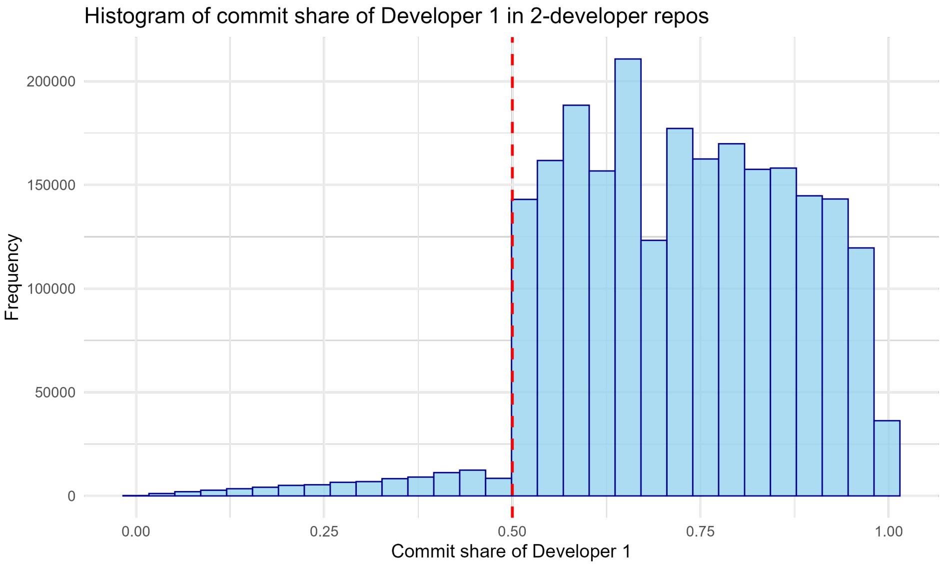
*Signif. Codes: ***: 0.01, **: 0.05, *: 0.1*

Collaborative projects with strong lead developers are the most successful





Lead developers commit much more



Distant teams are more successful

Dependent Variable:	Number of stars				
Model:	(1)	(2)	(3)	(4)	(5)
Variables					
Lead developer experience (log)	0.1035***	0.0530***	0.0140*	0.0164**	0.0258
	(0.0088)	(0.0081)	(0.0078)	(0.0078)	(0.0224)
Follower experience (log)	0.1561***	0.1017***	0.0992***	0.0997***	0.0999***
	(0.0072)	(0.0062)	(0.0060)	(0.0060)	(0.0145)
Distance (log)		0.2678***	0.2475***	0.2557***	0.2339***
		(0.0066)	(0.0066)	(0.0066)	(0.0121)
Lead developer commits (log)			0.5487***		
			(0.0170)		
Follower commits (log)			-0.4102***		
			(0.0183)		
Total commits (log)				0.3066***	0.3075***
				(0.0187)	(0.0314)
Lead developer commit share				2.598***	2.305***
				(0.1296)	(0.2255)
Fixed-effects					
language_time_fe	Yes	Yes	Yes	Yes	Yes
Fit statistics					
Observations	1,637,399	1,636,722	1,636,722	1,636,722	614,681
Squared Correlation	0.00687	0.01269	0.02378	0.02028	0.02549
Pseudo R^2	0.14658	0.20801	0.25819	0.24941	0.26197
BIC	22,526,679.5	20,904,939.1	19,584,430.7	19,815,448.8	7,717,733.2

Clustered (language_time_fe) standard-errors in parentheses
*Signif. Codes: ***: 0.01, **: 0.05, *: 0.1*

Counterfactuals

Counterfactuals (for future work)

Estimate γ, ν, ϕ_H, ϕ_L, τ_{12}, d_{12} and $F(z)$ structurally.

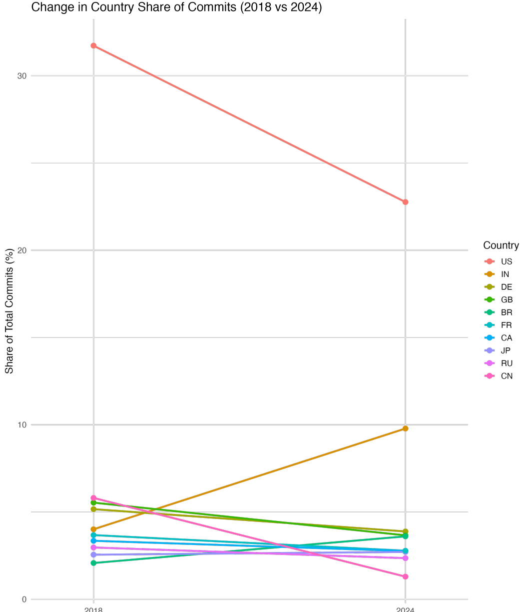
Study the impact of

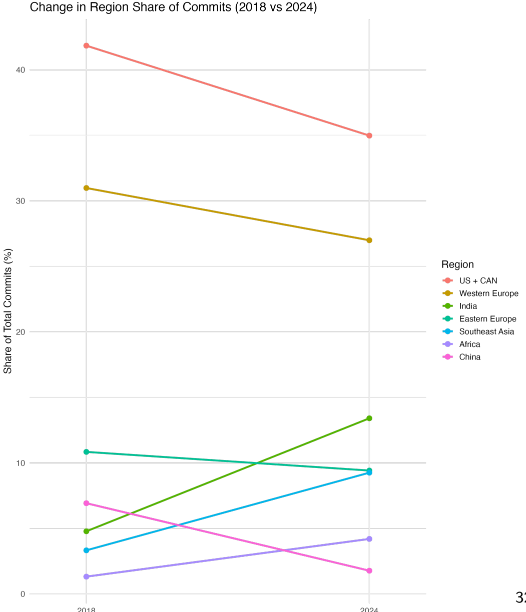
1. frictions in digitial services
2. generative AI

on

1. global software output
2. regional distribution of developers
3. inequality among developers

The Global South is increasingly important





Conclusion

Conclusion

Model of OSS where skill and distance determine team formation, collaboration, and project success.

Distant developers are less likely to join projects, but if they do, they are positively selected

Model quantifiable for policy analysis.

References

References

Atkin, David, M Keith Chen, and Anton Popov. 2022. "The Returns to Face-to-Face Interactions: Knowledge Spillovers in Silicon Valley." National Bureau of Economic Research.

Eaton, Jonathan, and Samuel Kortum. 2002. "Technology, Geography, and Trade." *Econometrica* 70 (5): 1741–79.

Fackler, Thomas, and Nadzeya Laurentsyeva. 2020. "Gravity in Online Collaborations: Evidence from Github." In *CESifo Forum*, 21:15–20. 03. München: ifo Institut-Leibniz-Institut für Wirtschaftsforschung an der ….

Freund, Lukas. 2022. "Superstar Teams: The Micro Origins and Macro Implications of Coworker Complementarities." *Available at SSRN 4312245*.

Head, Keith, Yao Amber Li, and Asier Minondo. 2019. "Geography, Ties, and Knowledge Flows: Evidence from Citations in Mathematics." *Review of Economics and Statistics* 101 (4): 713–27.

Herkenhoff, Kyle, Jeremy Lise, Guido Menzio, and Gordon M Phillips. 2024. "Production and Learning in Teams." *Econometrica* 92 (2): 467–504.

Hoffmann, Manuel, Frank Nagle, and Yanuo Zhou. 2024. "The Value of Open Source Software." https://doi.org/10.2139/ssrn.4693148.