The Global Software Production Network

Carlo Birkholz*

David Gomtsyan†

December 10, 2025

Abstract

Can developing countries benefit from exporting opportunities in the growing sector of tradable services, given the near free information flow via the internet and wage differentials relative to developed countries? Focusing on the software development industry, we analyse data from 2.68 million software projects across 5,200 locations, and estimate an economic geography model in which locations trade tasks. The results reveal three factors limiting exports: (i) significant productivity differences within and between countries; (ii) a notable decline in trade volumes with distance; (iii) sorting patterns among software developers that are suggestive of brain drain.

Keywords: Productivity, IT, services trade, migration, sorting.
JEL code: F1, L86, O15.

*University of Göttingen, ZEW Mannheim; e-mail: *c.birkhlz@gmail.com*.

†Centre de Recerca en Economia Internacional (CREI); e-mail: *dgomtsyan@gmail.com*.

We would like to thank Rüdiger Bachmann, Andreas Fuchs, Dávid Nagy, Èric Roca Fernández, Todd Schoellman, and Mu-Jeung Yang for valuable comments.

1 Introduction

Over their development path advanced economies have experienced a substantial increase in the share of the high-skilled services sector. In the US, the share of high skilled services exceeds 50% of total value added (Buera and Kaboski, 2012). Notably, many segments within this sector produce tradable output. Given that technological advances of recent decades reduced the cost of digital information flows to near zero, new exporting opportunities may arise for developing countries, where wages are lower than in developed countries. Are developing countries in a position to take advantage of these opportunities? We address this question by employing novel data that allow us to study the global software development industry, one of the fastest evolving parts of the high-skilled services sector.

Our main analysis is based on GitHub data from 2.68 million projects and 2.75 million users, and their interactions. The available data allow us to observe the locations of users at the city level, their contributions to specific projects, as well as their follower networks. We employ this information to construct flows of software code between locations from project level collaborations. Based on these flows, we propose a spatial model in the spirit of Eaton and Kortum (2002), in which software developers in different locations trade in tasks. By estimating the gravity equation derived from the model, we recover distance elasticities and productivity parameters at the city level.

According to our estimations the San Francisco Bay Area emerges as the unambiguous leader, followed by other cities located on the West Coast of the US. Among developing countries, the most productive locations are Ho Chi Minh City in Vietnam, Bengaluru in India and other prominent IT-cities in China, India and Eastern Europe. Overall we find that there is a tight relationship between our measure and GDP per capita at the country level, and between per capita nighttime luminosity at the city level. We also find that estimated productivity differences in the software industry between the richest and poorest countries are comparable or even larger than those derived from macro data encompassing broad sectors. This means that the poorest countries are performing worse in the production of software code than in the production of goods and other services. Moreover, we construct a separate productivity measure for the generation of final software products, which presents a higher value activity than provision of coding services. We find that the comparative advantage in the generation of final software products relative to coding services increases with GDP per capita.

Despite the fact that, from a technological perspective, there are no spatial frictions to the trade in software code, our gravity equation estimates imply that distance has a negative effect on trade volumes. Specifically, our estimated distance elasticity is in the range of 0.7-0.9, which is comparable in size to the value of 1 obtained for the flow of goods within the US (Allen and Arkolakis, 2018). To shed light on this sizable effect, we examine three types of mechanisms. First, we ask whether it simply reflects technological specialization across space. Constructing language-specific code flows and re-estimating the gravity equation with programming-language fixed effects leaves the distance elasticity essentially unchanged, indicating that the result is not driven by the spatial composition of technologies. Second, we estimate distance elasticities by programming language and find systematic heterogeneity: elasticities are largest (in absolute value) for content-presentation and web languages and smallest for systems and cloud languages, consistent with stronger local demand and context-specific coordination needs in the former. Third, our preferred interpretation of this sizable effect is that distance affects the movement of people, and the networks in which they collaborate. The production network is shaped by collaborations formed through in-person interaction, such that online software production cannot be understood as a process that operates independently from offline location. We corroborate this hypothesis using domestic flight data from the US. We include the number of passengers between a pair of cities as a control variable in our gravity regressions and find that it is positively related with code flows.

We then investigate the migration patterns of IT specialists within and across countries. In our data we observe the location of these software developers at different points in time. We construct a proxy for the quality of their skill set based on the centrality of the software developer in the follower network of all GitHub users, which we derive through the recursive ranking algorithm PageRank. We document that there are strong sorting patterns of migration both within and across countries based on this quality proxy. For example, we observe that IT specialists who are ranked higher in a city at time t are more likely to migrate to a more productive city (or a country with higher GDP per capita) in period $t + 1$. We further show that immigrants tend to have higher quality than the median resident in the destination. These results hold both when migrants move to places that are rated higher in terms of IT productivity than their origin location, and when they move to countries with a higher GDP per capita than their country of origin.

We develop the migration dimension further by implementing a decomposition exercise that separates intrinsic location-specific productivity from individual productivity. Because this exercise requires time varying productivity measures, our sample shrinks substantially, so we interpret the results with caution. The decomposition indicates that almost the entire difference in city productivity is driven by individuals, i.e., productive individuals selecting into specific locations. We note that such selection can be driven by amenities, which opens room for policy action.

Taken together, our results suggest that – barring effective policy interventions – developing countries are unlikely to reap large benefits from software code exports for three reasons: First, the ability to export requires high productivity. However, our estimates show that the productivity gap in the software development sector between rich and poor countries is of a magnitude comparable or even larger to the gap in the service sector or manufacturing. Second, our estimates show that there are substantial spatial frictions which hamper trade flows. Third, the migration patterns we document indicate that developing countries experience a brain drain, which may make it harder to catch up with the technological frontier.[1]

We validate our data in several steps. First, we use two alternative approaches to measure the role of each location in the software production process. As one alternative, we construct a graph of locations in the world which are linked to each other by their observed software code flows. We again apply PageRank to recursively determine the centrality of each node (location) in the graph. As another alternative, we aggregate the individual scores we obtained from applying PageRank to the follower network at the level of locations. The results obtained according to both of these alternative approaches are closely correlated with the productivity measures obtained from the structural estimation. Second, we validate our measure for the US sub-sample by regressing it on wages of US IT specialists obtained from the American Community Survey at the location level, and for the full sample by regressing it on wages of IT specialists globally from the Stack Overflow Developer Survey at the country level. We find an economically large and statistically strong relationship. Third, we construct university rankings for the US, the UK and Germany based on individual software developers' quality scores and their reported affiliation. The list shows close resemblance with conventional rankings, such as by US News or the Academic Ranking of World Universities.

For the analysis of the questions we pose, GitHub data have important advantages over the patent data that have been widely used in the literature. First, they cover a wide range of countries with varying levels of GDP per capita, and capture an extensive membership and activity network in many developing countries, whereas the literature based on patents has focused on a small set of high income countries. Second, we observe activities at high frequency levels, while patenting is a relatively rare activity, especially at the individual level, and many inventors register only one patent during their lifetime. This makes the analysis of inventor migration complicated because economists observe inventors' locations only when they reg-

[1] If migrants also facilitate the diffusion of knowledge to their home countries, then the negative effects of brain drain would be less severe. We are silent on this channel.

ister a patent, so they need to observe the same inventor registering patents in different locations to document an event of migration.[2] Third, in the GitHub data joint participation in projects by members located in different locations is more common, which enables us to study interactions across space. Finally, software production is relatively less dependent on the investment of physical capital than other high skilled sectors, and members of teams are less confined by physical distance; they do not need to be located in laboratories with special equipment. Thus, our setting allows us to focus on the human capital and human interaction aspect of the innovation process. A drawback of our data is that we measure bilateral collaboration by counts of commits which does not account for differences in code quality or task complexity. This is driven by data availability. We mitigate this limitation by constructing an individual-level proxy of quality and show that our analysis is robust to adjusting trade flows with this measure.

There is a large literature that tries to measure productivity levels across countries (see, for instance, Klenow and Rodríguez-Clare, 1997; Hall and Jones, 1999). Methodologically we follow Waugh (2010) and use a trade model to recover productivity parameters. In contrast to the aforementioned papers we focus on one industry, but our productivity measures are at the city level rather than at the country level. Within this literature, it is worthwhile emphasizing papers that specifically focus on the level of human capital. Since software production is human capital intensive and individuals can provide their services to firms in distant locations, we believe that the human capital component in our productivity measure is large. However, it cannot be interpreted as being a measure of human capital exclusively, because other factors, such as agglomeration forces acting at the city level, are also included in our estimated productivities. Given the difficulties related to the measurement of schooling quality, researchers have used wages of migrants in destination countries to measure human capital

[2] For example, in the dataset used by Akcigit, Baslandze, and Stantcheva (2016) 52% of inventors have only one registered patent. For this reason the authors base their analysis only on top inventors who register patents frequently.

(Clemens, 2013; Hendricks and Schoellman, 2018; Martellini, Schoellman, and Sockin, 2024). In this literature, researchers rely on wages to obtain measures of worker quality. However, when transitioning from one location to another, workers may face imperfect transferability of skills, discrimination, or lack of local networks. All of these factors can lead to lower estimates of migrants' true skills. Because our measure is not based on wages, it is less likely to be affected by those factors. Our setting of open-source collaboration is particularly well suited as it reduces institutional and organizational frictions further: contributors can demonstrate and exchange code without employment contracts or work permits, and firms can source solutions globally without establishing a local subsidiary.

We also contribute to the literature on trade in services. The decline in communication costs has led to an increase in services trade (Eckert, 2019). However, a lack of data makes it difficult for researchers to measure the extent of such trade flows. Eaton and Kortum (2018), using 2010 international bilateral trade data, find a distance elasticity of 1.4 in professional services and administrative services. Other studies combine structural models with industry employment data from the US to generate trade in services without observing the actual flows (Gervais and Jensen, 2019; Eckert, 2019). Hsieh and Rossi-Hansberg (2023) study trade in non-tradeable services through the expansion of affiliates.

We also relate to other papers and emerging work using GitHub as a data source. Wachs, Nitecki, Schueller, and Polleres (2022) utilize the geolocation of software developers on GitHub to document the spatial distribution of software developers between and within countries. Wright, Nagle, and Greenstein (2023) show that greater participation in open source development on GitHub at the country level leads to an increase in the number of new technology ventures in subsequent years. Wachs (2023) investigates brain drain as a consequence of conflict by following the migration of software developers on GitHub after the onset of the Russian invasion of Ukraine. Like us, Fackler, Hofmann, and Laurentsyeva (2023) - who study collaboration in remote teams around the COVID-19 pandemic - also use

GitHub data and estimate gravity equations at city-pair level. Their estimated distance elasticity coefficients are below 0.5, which are smaller than ours. There are, however, some key differences in our estimations related to sample selection, data construction, and the estimation specification that explain the differences in the estimated elasticity.[3]

We structure the remainder of our paper in the following way: We describe the features of GitHub data and complementary data sources in Section 2. In Section 3 we lay out our spatial equilibrium model and alternative approaches to calculate city-level productivities. We then present the results of our estimations and relate them with GDP per capita in Section 4. In Section 5 we study the migration patterns of software developers. Section 6 concludes.

2 Measuring trade in services with GitHub data

Our primary data source is a snapshot of the universe of GitHub users and their public activity on the platform in March 2021. This data is the latest available version of the GHTorent project that periodically mirrors GitHub's public event timeline through GitHub's API (Gousios, 2013). We supplement this with a snapshot of the data from June 2019 from the same source to identify changes in the reported location of users to study migration patterns.

GitHub is a service for software development and version control. It is the dominant service for hosting open source software.[4] One of the main advantages of GitHub compared with other version control solutions is that it accommodates large teams of developers working independently. As a

[3]They choose to drop locations below an arbitrary size threshold yielding only around 700 locations and appear to be using the entire sample of GitHub users with any location reported. As we discuss in section 2, we apply a number of careful data cleaning steps to the reported locations of users, to avoid introducing bias from systematic errors in the geocoding of the locations. Finally, we estimate the gravity equation with importer and exporter fixed effects at the location level, rather than their choice of country level, which is more appropriate to address multilateral resistance (Fally, 2015).

[4]"What is GitHub?" The Economist, Jun 18, 2018.

result, most widely used open source software programs have repositories on GitHub. It is also worthwhile to note that, despite being open source, most popular programs with many users are owned by large organizations and generate revenues.[5] Some widely known names are Linux, MySQL, and Firefox. Owners of these products rely on various business models to generate revenues; the most common revenue generation model is to sell enterprise versions or additional bundles that complement the free version. Since these are sophisticated and advanced products, the owners frequently hire professional software engineers for further development and updating.

Users There are a total of 45.8 million registered users in the 2021 data snapshot; these users can be uniquely identified based on their ID and user names. Registered users are mostly individuals, but can in some instances also be organizations, which are identified through a user type variable. The range of engagement and activity on the platform varies widely, as well as the completeness of the user profiles. We observe around 3.7 million users with some degree of information about their physical location. Locations are self-reported in a free text field; this information is automatically translated into a geolocation (longitude and latitude). We undertake rigorous cleaning efforts to ensure that the user input is reasonable, and that the automated geocoding is accurate. As a first step in this cleaning effort, we drop users reporting locations such as '*the internet*', '*the world*', '*anywhere*', '*remote*', '*future*', '*darknet*', '*404*', '*Earth*', '*Moon*', '*universe*', '*galaxy*', '*Milky Way*', '*Pluto*', '*Mars*', or '*space*'.[6] In a second step, we drop all users with location information that is not granular enough to map them onto cities accurately. This is crucial, as users reporting information on the country level, for example, receive the geocoordinates of the country's capital. As a third step, we manually review common user entries that represent over 1% of the observations at each location, excluding the smallest 1% of locations.

[5] Commercial open-source software company index.

[6] We manually inspect location names containing these strings to not lose valid addresses such as *Moon Vista Avenue, Las Vegas*.

This process allows us to eliminate any remaining significant errors in user allocation. We are left with a sample of 2.75 million users with cleaned locations, which is the subset of data we employ whenever our analyses rely on location information. Figure F1 in the appendix plots all unique user locations across the world. In terms of the selection of users indicating their location, we are confident that our sample reflects the active, professional users of the platform, as professional use of the platform incentivises a fully completed profile to facilitate communication and work opportunities. We provide an extended discussion of the representativeness of our sample in the Appendix section A.3.

For the time period up to 2019 we observe an additional aspect of the social network within GitHub, namely the followers and following of each user. The following functionality is an important feature for collaboration on the platform, as it enables users to get directed updates on other users' activities, such as changes made within shared projects or new projects started.[7] Around 3.8 million users follow at least one other user, and those who follow at least one person follow an average of 7.8 users.

Projects We observe over 189 million projects in the database, which are uniquely identified by project IDs. GitHub projects are organized into repositories, which contain all of the contents of a specific project; in the following, we will use the terms "project" and "repository" interchangeably. We link users to projects via the unique project IDs. Every project has one owner, who typically holds a central role within the project, as we demonstrate in Appendix B, and users who – conditional on taking part in any project – belong on average to 4.5 projects. Whenever we study collaboration within projects based on geographic location, we define the projects'

[7] For instance, in a forum post discussing the following functionality on GitHub, users write "[...] when I find someone contributing to a library I use or a project that does what I need, I want to know about it immediately. [...] I follow the core developers of some of the main business critical libraries that we use (and sometimes their upstream dependency projects) so I can get a heads up on any potential breaking changes coming down the line" and "The same reasons you follow anyone on social media- to see what they're doing".

origin as the owner locations. Given that we do not observe locations for all users, as discussed above, these analyses rely on a subsample of 48.5 million projects for which owner location information is available. When constructing flows of code between locations in a project, we additionally require information on the locations of the contributing users. For 2.68 million projects we observe the location of the owner and the location of at least one project contributor.

Commits Commits are the primary user action to advance a project. They refer to a version of changes made to a repository's files. Changes to a project that are initially made locally are grouped and pushed to update the online version of the project. Commits typically come with a short message describing changes made, so that one can keep track of file versions. For each commit we identify the author, the committer and the project owner. The author and committer can be different users, for instance when users who are not project members suggest changes; a process explained further in the section **Forks and pull requests** below.[8]

In our analysis we construct flows of software production based on authors and owners. We clean the commits data in two main ways before constructing these flows: First, we do not consider commits where the author and owner are the same user – a construct we term self-links. Second, we alleviate potential biases stemming from bot activity by dropping users that are tagged as 'fake' by GitHub and by dropping commits that resemble the automated nature of bot activity. For the latter we construct the within-project variance of the commit frequency of users with at least 25 commits, and drop them if they display a variance of zero, which means they commit in exactly steady intervals.[9]

We then define X_{ij} as the volume of code that flows from location j to location i determined by the following expression:

[8]Another instance can occur when multiple project members collaboratively work on the same project branch (part of the project) and only one of them commits the others' changes.

[9]Bots are software that run reoccurring tasks in an automated fashion.

$$X_{ij} = \sum_{k \in K} commits_{jk} \times 1[owner_{ik} = 1], \tag{1}$$

where K is the set of projects, $commits_{jk}$ is the number of commits on project k by users from location j and $1[.]$ is the indicator function equal to 1 if the owner of project k is in location i. Intuitively this means that the volume of code flowing from location j to location i is the sum of commits from location j in projects whose owner is located in i. In Appendix B we provide motivation for this approach and discuss alternatives.

Forks and pull requests Users may copy projects, in GitHub terminology "fork", and create modifications or build a different version of the parent project. There are two main rationales for doing so: First, a user may fork a project, modify it and then propose to merge the changes with the main project – an action that is referred to as creating a pull request. If accepted the changes are committed to the original project. Second, a user may create a new independent software, which uses the original software as an input. In this case the fork represents an import of final software product.[10] We treat commits generated via pull requests equivalently to commits within a repository described above. "Forks" that do not generate upstream commits are classified as trade in final software, not as intermediate collaboration. While our paper focuses on trade in services that is represented by the gradual contribution of commits to the development and improvement of a software product, the trade in final software products or ideas captured by this second category of forks is an additional interesting aspect of the global software production network. For the remainder of the paper we will use the terms trade in services and trade in ideas/final software for these two dimensions of trade activity on GitHub. We empirically investigate trade in ideas in Section 4.4 noting however the caveat that the volume of transactions is much lower than for trade in services implying a noisier measure.

[10]By final software product we mean a software product which can be used either by consumers or by other software developers as an input for the production of other software.

Other data In addition to the GitHub data, we use geographic information on functional urban areas (FUAs) and administrative regions, population and nighttime luminosity data from satellite images, US air traffic and the location of US airports, as well as income data at sub-national and country level. We describe the construction of all auxiliary data we employ in the Appendix section A and in Table A1 we list all samples we use throughout the paper.

3 Methodology

We propose several approaches to determine the productivity of each city in the global software production process. Our main approach is based on the standard Eaton and Kortum (2002) model in which individuals in different locations produce and sell software code. This model allows us to derive a structural gravity equation and recover productivities of locations. Then, we propose two alternative reduced-form approaches for ranking cities. While each approach has its unique up- and downsides, we find that they produce consistent results.

3.1 A model of trade in tasks

The model is based on the standard Eaton and Kortum (2002) framework. Several papers have used this framework to impute country-specific productivity parameters (Waugh, 2010; Levchenko and Zhang, 2016). We follow the approach used in these papers to impute the level of software development productivity in specific locations. In our setting, trade takes place in software development services or tasks. We focus only on this sector and do not describe the rest of the economy. To the extent that we are interested in estimating distance elasticities and productivities for software development, the weight of software in household preferences or its contribution as an input to other sectors does not matter (see Levchenko and Zhang, 2016). The only assumption we need is that labor is the only input required to produce software code. This would not appear to be a very strong assumption because in the software development process the share of labor is likely to be higher than in most other industries. Moreover, software development tools (programs and cloud services), which are probably the next most important input, are either available as open source or highly tradeable without much variation in prices across space.

The analytical formulation of the problem is similar to the above mentioned papers. However, given the nature of our data and the environment of open source software production, we provide somewhat different interpretations. In particular, in a conventional trade model, the unit of production is a firm located in location i that produces a differentiated good q with efficiency $z_i(q)$ by hiring labor (inputs). In our case the unit of production is an individual rather than a firm, and this individual uses his or her own labor. We assume that software developers are endowed with a fixed amount of time which they allocate to solving open source problems. In our context, the differentiated good is a specific segment of the overall code. The solutions submitted by developers require proofing and potentially additional improvements or tuning from the owner of the code, which takes owners' time. The amount of time required to improve the proposed solution is inversely proportional to the productivity $z_i(q)$ of the developer who submitted it. Additionally there is an iceberg trade cost d_{ij}. An interpretation of this cost is that the developer with productivity $z_i(q)$ may lack familiarity with a given project, as a result of which the quality of his proposed solution decreases by a factor of d_{ij}. Familiarity with a project can be built through interactions, the likelihood and intensity of which decrease with physical distance. Thus, the code owner will adopt the best solution, or equivalently the solution proposed by the developer with the highest $z_i(q)$ adjusted by the iceberg cost ($min_{i=1,...,N}\{\frac{d_{ij}}{z_i(q)}\}$).

Individual productivities are drawn from the Fréchet distribution with the cumulative distribution function $F_i(z) = e^{-T_i z^{-\theta}}$. We allow the parameter T – which governs the average of the productivity draws – to be location-specific; this is our main object of interest. We interpret it as the average level of software development productivity or skills in location i. Higher values of T_i imply higher levels of average productivity. θ captures the dispersion of productivity draws.

The final software is produced using a CES production function that aggregates a continuum of task varieties $q \in [0, 1]$ according to the following formulation

$$Q_i = \int_0^1 \left[Q_i(q)^{(\epsilon-1)/\epsilon} dq \right]^{\epsilon/(\epsilon-1)},$$

where ϵ denotes the elasticity of substitution across varieties q and $Q_i(q)$ is the amount of variety q that is used in production. Following the steps in the aforementioned literature, the fraction of software development services provided (open source problems solved) by location j in the share of total software services consumed in location i is given by the following gravity equation

$$\frac{X_{ij}}{\sum_j X_{ij}} = \frac{T_j (d_{ij})^{-\theta}}{\Phi_i},$$

where $\Phi_i = \sum_j T_j (d_{ij})^{-\theta}$ is the multilateral resistance term. Dividing X_{ij} by the analogous expression for X_{ii} and taking logs, we obtain the conventional gravity equation

$$ln \left(\frac{X_{ij}}{X_{ii}} \right) = ln \left(T_j \right) - ln \left(T_i \right) - \theta ln(d_{ij}), \tag{2}$$

where X_{ij} denotes the volume of the flow of goods from location j to location i, the construction of which was described in equation (1). Next we express the log distance cost from equation (2) as

$$ln(d_{ij}) = d_k + a_{ij} + b_{ij} + Lang_{ij} + im_i + \nu_{ij},$$

where d_k is the contribution to trade costs of the distance between i and j measured in miles. Other variables are an indicator if cities are in the same country (a_{ij}), an indicator if countries share a border (b_{ij}), an indicator for a common language $Lang_{ij}$ and an importer fixed effect im_i. Substituting the expression for trade costs back to the equation (2) we obtain

$$ln\left(\frac{X_{ij}}{X_{ii}}\right) = \underbrace{ln\left(T_j\right)}_{\text{Exporter FE}} \underbrace{-ln\left(T_i\right) - \theta im_i}_{\text{Importer FE}} \underbrace{-\theta d_k - \theta a_{ij} - \theta b_{ij} - \theta Lang_{ij}}_{\text{Bilateral observables}} - \theta \nu_{ij} \tag{3}$$

In equation (3) the first term captures exporter fixed effects, which is the main object of interest. We estimate equation (3) using PPML. As a result of the estimation we obtain exporter fixed effects for each location, which have the following relationship with the productivity parameter

$$exp(EFE_j) = T_j, \tag{4}$$

where EFE_j are the exporter fixed effects from equation 3. An important detail is the inclusion of the term im_i in equation 3. An alternative approach to that is to include a term for exporters ex_j and use importer fixed effects to recover productivities from equation 4. There are four reasons that motivate our choice of the former over the latter approach. First, Waugh (2010) shows that including a term ex_j in equation 3 implicitly assumes that unit costs of production are the same across locations. In his context, this assumption is reasonable because higher productivity locations tend to have higher wages, so both forces push in opposite directions and counterbalance each other. Given the nature of open source contributions we have assumed that developers dedicate a fixed amount of time to solving open source problems without receiving a monetary compensation, thus the counterbalancing effect that operates through wages is not present. Hence, our preferred approach is to estimate equation 3 with the term im_i, which implies that unit costs are lower in more productive locations because they are more efficient. Second, the specification ex_j implies that locations face different exporting costs, in addition to the gravity terms for which we control. In the case of trade in goods, this friction can be justified by the quality of infrastructure such as ports, which is typically lower in developing countries.

In the case of software code, the role of these factors is arguably less important. Third, the ex_j approach requires information on the wages of software developers in cities around the world, for which precise data is not readily available. Fourth, by estimating equation 3 we recover a much larger number of fixed effects than with importer fixed effects. This is driven by the fact that there are more contributors (exporters) in the data than project owners (importers), which enables us to generate more variation for the identification of exporter fixed effects. Given these arguments, we prefer the use of exporter fixed effects; however, we demonstrate in Appendix section C that our productivity estimates are highly correlated with a specification using importer fixed effects and imputed city-specific wages.

3.2 Reduced form approach

Approach 1: Page rank algorithm. We think of locations as nodes of a graph and of X_{ij}'s as the strength of the links between nodes of the graph. The position of a node in a graph depends not only on its bilateral links but also on the links of the nodes to which it is connected, and so forth. In other words, the centrality of each node is determined recursively. A widely used approach for determining the centrality of nodes is the Page Rank algorithm (Brin and Page, 1998). The scores of locations are obtained as a solution to the following equation:

$$\begin{bmatrix} Score_1 \\ Score_2 \\ \vdots \\ Score_N \end{bmatrix} = \begin{bmatrix} (1-d)/N \\ (1-d)/N \\ \vdots \\ (1-d)/N \end{bmatrix} + d \begin{bmatrix} l_{11} & l_{12} & \dots & l_{1N} \\ l_{21} & \ddots & & \dots \\ \dots & & l_{ij} & \\ l_{N1} & \dots & \dots & l_{NN} \end{bmatrix} \begin{bmatrix} Score_1 \\ Score_2 \\ \vdots \\ Score_N \end{bmatrix} \tag{5}$$

where d is a parameter and l_{ij} is obtained by normalizing X_{ij} ($l_{ij} = \frac{X_{ij}}{\sum_j X_{ij}}$). The normalization ensures that $\sum_{i \in N} l_{ij} = 1$. If city i has no contributor involved in any project with other cities, then $l_{ij} = 0 \; \forall j$. Links to the node itself are not counted $l_{ij} = 0$ if $i = j$. Note that the resulting matrix, which is referred to as the adjacency matrix, is not necessarily symmetric. Equation 5 is solved by making an initial guess ($Score_i = 1/N$) and then making iterative computations until it converges. Typically, convergence is obtained rather quickly, which also turns out to be the case in our application.

Approach 2: Follower-based ranking As we described when introducing our data, on GitHub users may follow other users. The notifications received about followed users' public activities on GitHub enable and ease interaction. At the same time, people who make important contributions, generate new ideas or manage large projects are more likely to attract followers. We employ follower information to construct a graph in which each user is a node and directional edges between nodes are based on the following and follower links of users. We then apply the same recursive ranking algorithm described above to calculate the centrality score of each user. We interpret this measure as a proxy for individual quality. Conceptually being more central in the network of followers is likely highly correlated with individuals' quality, as the more and better work you do in projects, the more likely it is for others to follow you and receive updates on your work. In order to measure productivities at the location level we aggregate individual scores. Additionally, we use individual level scores to study the pattern of positive selection into migration in Section 5.1.

4 Results

In this section we begin by estimating the distance elasticity implied by the gravity equation and discussing three mechanisms behind the result, namely the role of in-person meetings, local demand and technological specialization. We then present location-level productivity estimates, relate them to nightlights per capita at the location level and GDP per capita at the country level, and compare estimated productivity gaps between rich and poor countries with macro data. We conclude the section by contrasting trade in tasks with trade in ideas.

Table 1: Distance elasticities for trade in tasks

	(1) X_{ij}/X_{ii}	(2) X_{ij}/X_{ii}	(3) X_{ij}/X_{ii}	(4) X_{ij}/X_{ii}	(5) $\hat{X}_{ij}/\hat{X}_{ii}$
Log distance in miles	-0.8222***	-0.7366***	-0.9061***	-0.6876***	-0.7319***
	(0.0454)	(0.0573)	(0.0823)	(0.0596)	(0.0071)
Controls	Yes	Yes	Yes	Yes	Yes
Same location dummy	No	No	No	Yes	No
Sample	FUA + Admin	FUA only	US FUA only	FUA + Admin	FUA + Admin
Observations	16,370,291	5,362,824	61,680	16,370,291	13,101,734
Pseudo R-squared	0.6196	0.6551	0.8424	0.6222	0.4954

Notes: Estimation results of equation 3. In columns (1), (4) and (5) the sample consists of all FUAs and Admin-2 regions. In column (2) we restrict the sample to FUAs, and in column (3) to FUAs in the United States only. In column (5) we multiply each commit by the individual quality measure of the author obtained from approach 2, in order to get a quality weighted trade flows ($\hat{X}_{ij}$). We winsorize this measure at the 99.95 level to account for extreme values produced by rare very small values in the denominator because of this multiplication. Controls include binary dummies for the same country, shared borders and shared official languages. Column (4) additionally includes a same location dummy. All specifications are estimated with PPML, and include importer and exporter fixed effects. * (**) (***) indicates significance at the 10 (5) (1) percent level.

4.1 Structural estimation results

Table 1 reports PPML estimates of the gravity equation. The estimated distance elasticity is around 0.8, which is close to the absolute value of the estimates for trade in goods (Allen and Arkolakis (2018) obtain a value of 1 for the US). This large estimate implies that geography continues to play an important role for trade in tasks, even though the flow of services between locations would seem to be frictionless. Our preferred explanation for this observation is that trade flows are determined in part by offline interactions involving in-person meetings, discussing ideas and making decisions on collaborations. Online software production does not occur in a vacuum, but is shaped by offline interactions. Thus, even though new technologies and platforms such as GitHub facilitate collaboration, they do not fully replace in-person interactions; rather they complement them.

In columns (2)–(4) of Table 1 we report several additional estimations to ensure the robustness of the result. In the second column, we restrict the sample to FUAs and construct the bilateral flows by ignoring users located outside FUAs. The estimated coefficient decreases slightly. In the third column, we restrict the sample to US FUAs only. The estimated distance elasticity increases slightly, suggesting that US domestic trade patterns do not vastly differ from global patterns. In column (4) we add a dummy variable for the same location. We expect the absolute value of the distance elasticity estimate to drop, because such pairs have 0 distance and interact with each other more intensively for instance because of better transport infrastructure.[11] However, the coefficient remains sizable.[12]

One limitation of our data is that our flow variable is constructed based on counts but there might be a substantial level of heterogeneity between commits. This is also an inconsistency with our model, as in the model trade volumes are measured in revenue not in quantities. To address this limitation, we multiply the commits made by individual j by their quality score, which we introduced in Section 3 under *Approach 2*. We denote the quality adjusted trade flows by $\hat{X}_{ij}$. Ideally the quality measure would be at the level of a transaction/commit, but we do not have this kind of information. Our assumption is that higher quality software developers make more valuable commits. Column (5) of Table 1 presents the result for this quality adjusted measure. The absolute value of the distance elasticity is only slightly lower compared to the one in column (1).

Production structures and software code flows in some GitHub projects may deviate from the model structure, for instance projects associated with large, multinational firms. Unfortunately, we lack reliable firm identifiers to cleanly isolate such projects. As a second best, we stratify the code flows by the size of the project team of the repository they are committed to. We then re-estimate the gravity equation restricting the sample to commits in projects of small ($<$ 50 project members) and large ($\geq$ 50 project mem-

[11]For instance, Bordeu (2023) provides evidence that transportation infrastructure changes discontinuously at the municipality boundaries.

[12]We also estimate the gravity equation using the June 2019 snapshot of the data. The estimated distance elasticity is very similar to the baseline with a value of -0.979. We conclude that the Covid-19 pandemic does not systematically affect the finding. Additionally, we estimate versions of the gravity equation that control for clock-hour differences (i.e., the cyclical differences in the hour of the day, where a 24-hour difference equals 0), where the estimated elasticity coefficient is -0.947. This result provides reassurance that the measured distance elasticity reflects spatial frictions, rather than being confounded by temporal misalignments between more distant locations that exacerbate communication challenges.

bers) production teams. Table E1 in the Appendix shows that the absolute distance elasticity is smaller for large-team projects (≈ 0.6) and larger for small-team projects (≈ 0.9), consistent with the notion of tighter intra-firm coordination attenuating distance frictions.

4.1.1 Mechanism: In-person meetings

Our interpretation above stresses the importance of in-person meetings in driving the distance elasticity. This mechanism is consistent with the idea that trade is hindered not only by transportation costs but also by information frictions which increase with distance (Allen, 2014). These information frictions are understood to be potentially large in online goods markets, particularly with a large number of market participants (Bai, Chen, Liu, Mu, and Xu, 2022), and for trade in goods face-to-face meetings are an effective way to alleviate them (Startz, 2016). For trade in services, these frictions are likely exacerbated since product and quality details are often more difficult to define and verify, such that the information friction component in the trade costs may exceed that in goods trade.[13]

We probe this mechanism using US domestic air traffic data from the Bureau of Transportation Statistics. We augment the gravity equation with bilateral final-destination passenger and air-freight flows. Because flight connections lower the financial and time costs of in-person meetings, we expect a positive relationship between the volume of passenger traffic and trade volumes between locations. Column (1) of Table 2 confirms this expectation. A competing interpretation is that locations linked by shared industries both demand similar software inputs and generate passenger traffic. To assess this possibility, we implement a placebo test by replacing passenger traffic with air-freight volumes. However, column (2) of Table 2 shows no relationship between freight flows and software development task flows.

[13] While from an end user perspective it is perhaps easy to verify whether a software does what it should, from a software development perspective dimensions such as the efficiency and compatibility with existing and future code need to be considered in addition to functionality.

Table 2: Trade in task with air traffic controls

	(1) X_{ij}/X_{ii}	(2) X_{ij}/X_{ii}	(3) X_{ij}/X_{ii}	(4) X_{ij}/X_{ii}
Log distance in miles	-0.5512***	-0.5517***	-0.5512***	-0.5517***
	(0.1290)	(0.1290)	(0.1290)	(0.1289)
Passengers in million	0.0797***		0.0582***	
	(0.0213)		(0.0137)	
Freight in million		0.0003		-0.0005
		(0.0007)		(0.0062)
Controls	Yes	Yes	Yes	Yes
Same location dummy	Yes	Yes	Yes	Yes
Sample	US FUA + Admin	US FUA + Admin	US FUA + Admin	US FUA + Admin
Observations	625,260	625,260	625,260	625,260
Pseudo R-squared	0.6897	0.6897	0.6897	0.6897

Notes: Estimation results of equation 3 with added controls for the number of air traffic passengers in millions and air traffic freight measured by weight in million pounds. The sample consists of all US FUAs and Admin-2 regions. In columns (1) and (2), passenger and freight streams are calculated for the period January 2009 to April 2021, whereas in columns (3) and (4) for the period January 1990 to December 2008. Controls include binary dummies for the same country, shared borders, shared official languages and same location. All specifications are estimated with PPML, and include importer and exporter fixed effects. * (**) (***) indicates significance at the 10 (5) (1) percent level.

Another potential concern is reverse causality: successful collaborations could themselves induce travel. We address this by recomputing air traffic using only the pre-GitHub period (1990–2008) and re-estimating the specification in columns (3) and (4) of Table 2. The passenger effect persists, consistent with in-person meetings facilitating collaboration rather than vice versa.

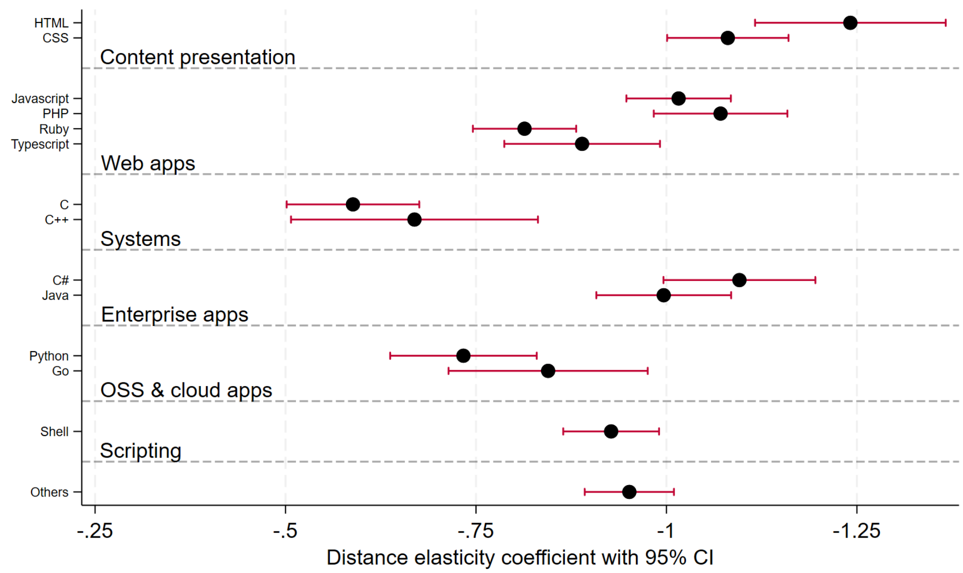
4.1.2 Mechanism: Local demand and technological specialization

Other potential mechanisms are technological specialization and region specific demand: if tasks tied to particular technologies require more local coordination with clients or domain experts, their spatial frictions should be larger. We explore this by re-estimating the gravity equation language-by-language for languages with more than 1,000,000 commits, assigning each commit to the main language of it's project.

Figure 1 plots the resulting distance elasticities and 95% confidence intervals. Two patterns stand out. First, *content-presentation* languages (HTML, CSS) exhibit the largest magnitudes (-1.08 to -1.24), consistent with the idea that tasks in these programming languages are close to end users, local content, and design teams and therefore requiring closer interaction or developers' knowledge of local preferences. Likewise, languages commonly used for building *web and enterprise applications* display larger distance elasticities (ranging from -0.814 to -1.096), consistent with a larger need for context specific knowledge. Second, *systems* languages (C, C++) have the smallest magnitudes (-0.589 to -0.669), in line with more modular, infrastructure-oriented work that is tradable across space and coordinated through standardized interfaces. Similarly *OSS & cloud* languages (Python, Go) lie on the lower end of estimated distance elasticities (about -0.734 to -0.845), reflecting wide applications and global communities, specifically for Python in data science, machine learning and AI development. While confidence intervals overlap across several languages, the general pattern is clear: languages more tightly linked to local content and client-facing coordination display larger distance elasticities.

We also pool all bilateral language-specific flows and re-estimate equation 3 including language fixed effects. The resulting distance elasticity is -0.764, nearly identical to our main estimate. We conclude that programming languages explain meaningful variation in line with expectations around a common underlying spatial friction that is not solely determined by technological specialization or regional demand.

Figure 1: Distance elasticities by programming language



Notes: Each point is the PPML estimate of the distance elasticity from equation 3, using only flows for commits to projects whose primary language is indicated on the y-axis; bars show 95% CIs. 'Others' *groups together commits in all languages with less than 1,000,000 commits. Languages are grouped by their common application.*

4.2 City productivities

Productivity measures for the top 35 cities constructed according to the methodology described in Section 3.1 are presented in the first column of Table 3. It is reassuring that San Jose, which according to our FUA definition includes the entire Bay Area, appears at the top of our ranking. The positions of Portland, nicknamed Silicon Forest with its substantial technological cluster, Bengaluru, the IT capital of India, and appearances of other well known IT hubs such as Ho Chi Minh City, Berlin and Prague lend further credibility to our results.

In columns 2 and 3 of Table 3 we present the results for the two reduced form approaches. One noticeable difference is that, for these approaches, the list is dominated by large cities. A key advantage of the structural model is that the results do not depend on city size. This can be seen from equa-

Table 3: Ranking of the top 35 cities across the world

Rank	Model	Approach 1	Approach 2
1	San Jose	San Jose	San Jose
2	Berlin	New York	New York
3	Seattle	Seattle	London
4	Portland (Oregon)	Boston	Beijing
5	Los Angeles	London	Seattle
6	London	Washington D.C.	Berlin
7	Ho Chi Minh City	Los Angeles	Shanghai
8	Tokyo	Paris	Los Angeles
9	Moscow	Berlin	Portland (Oregon)
10	Cape Town	Beijing	Tokyo
11	Nanjing	Tokyo	Boston
12	Oslo	Chicago	Paris
13	Sydney	Denver	Guangzhou
14	New York	Atlanta	Hangzhou
15	Seoul	Portland (Oregon)	Toronto
16	Beijing	Austin	Austin
17	Prague	Shanghai	Chicago
18	Shanghai	Amsterdam	Washington D.C.
19	Lisbon	Toronto	Denver
20	Dallas	Seoul	Bengaluru
21	Gothenburg	Philadelphia	Melbourne
22	Paris	Bengaluru	Pittsburgh
23	Melbourne	Guangzhou	Moscow
24	Mumbai	Tijuana	Vancouver
25	Bengaluru	Singapore	Stockholm
26	Austin	Zurich	São Paulo
27	Munich	São Paulo	Montreal
28	Singapore	Vancouver	Amsterdam
29	São Paulo	Stockholm	Sydney
30	Vancouver	Montreal	Philadelphia
31	Wellington	Sydney	Singapore
32	Tijuana	Cambridge	Atlanta
33	Zurich	Hangzhou	Madrid
34	Louisville	Delhi [New Delhi]	Seoul
35	Fukuoka	Moscow	Munich

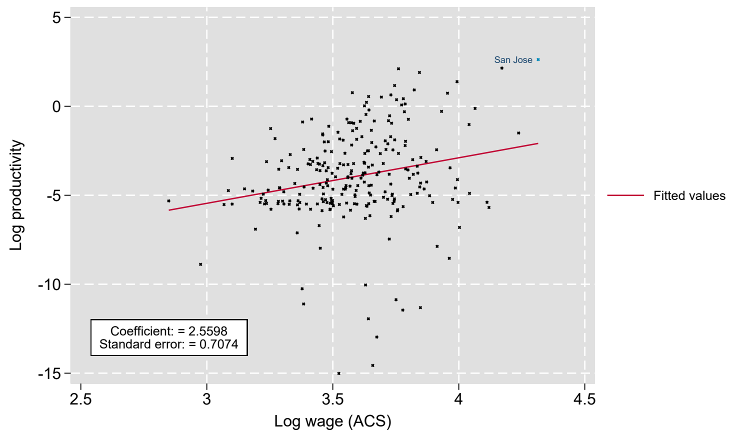
Notes: This table displays the top 35 locations ranked by the three different methodologies described in Section 3.

tion 3, where the outcome variable in the gravity equation is normalized by internal interactions. In the case of the recursive ranking approaches, on the other hand, it is natural that large cities receive more links; accordingly, it is not proper to interpret the scores obtained from these two methods as measures of productivity. The method based on the aggregation of individuals' scores can actually be interpreted as a proxy for total output.

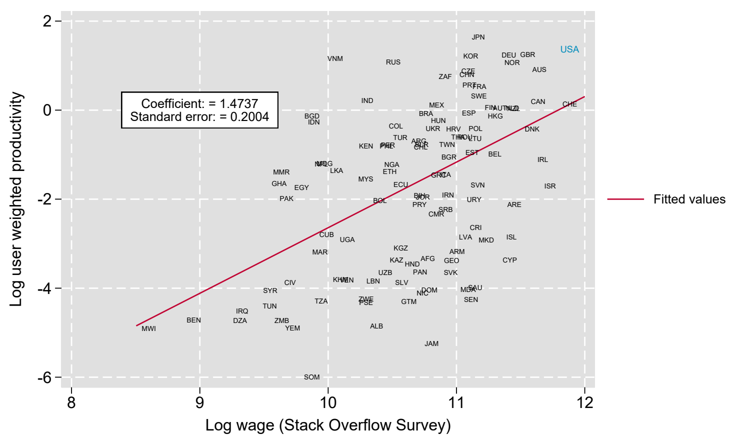
In the absence of direct measures of productivity, we formally validate our measure by using the wages of software developers, which are closely related to productivity, especially in an industry where the share of labor is high. We begin by restricting our sample to the US and regress our productivity measure on the wages of IT specialists in US cities. Our wage data come from the ACS, as described in Section A.2. The results are displayed in panel (a) of Figure 2. We observe that both variables move together, also indicated by a significant slope coefficient of 2.56. In panel (b) of Figure 2 we explore the relationship between our measure and the wages of software developers around the world. The wage data are constructed from a survey conducted by *Stack Overflow*. The data are at the country level, so we need to aggregate our productivity measures as well. To this end, we use the share of GitHub users of each location within each country and construct user weighted aggregate productivity at the country level. We restrict the sample to countries with multiple locations to reduce the influence of outliers, however the results are robust to using all countries. For this specification we also observe a positive relationship between our aggregated productivity measure and wages of software developers across countries. Clearly, the survey data have limitations, but both results together lend credibility to our estimated productivity measure. The advantage of the survey is that it covers many countries around the world, while the advantage of the US data is that they come from an official source and are less likely to suffer from selection bias. Additionally, in Appendix D we utilize information on software developers' affiliations to construct university rankings for the US, UK, and Germany and compare them with such rankings from other sources.

Figure 2: Estimated productivities and IT-sector wages

(a) US FUAs productivity and IT-related professions' wages



(b) User weighted productivity and IT wages country level



Notes: Panel (a) plots the relationship between log productivity estimated from the model and wages of IT specialists, constructed from the ACS, across FUAs in the US. Panel (b) plots the relationship between log productivity aggregated at the country level by applying user weights across locations within each country and wages of IT specialists from the 2023 Stack Overflow Developer Survey.

Table 4: Correlations between IT productivity and nighttime luminosity per capita and GDP per capita globally

	(1) Log productivity	(2) Log productivity	(3) Log productivity	(4) Log productivity
Log nightlights per capita	0.5547***			
	(0.0620)			
Log GDP per capita		0.8214***	0.8253***	0.8963***
		(0.1063)	(0.1145)	(0.1177)
Sample	FUA	Country level	Country level	Country level
Aggregation method		Average of top 5%	Population weighted	User weighted
Observations	2,608	122	122	122
R-squared	0.0286	0.3664	0.3663	0.3654
F	80.12	59.72	51.94	58.00

Notes: The dependent variables are log productivity estimated from the model. For the country level regressions productivities are aggregated using three different approaches: first, by averaging productivity in top 5% locations (column 2); second, by applying population weights in each location (column 3); third, by applying GitHub user weights in each location (column 4). For the country level regressions, we restrict the sample to those countries with multiple locations to reduce the influence of outliers, however the results are robust to using all countries. Standard errors are robust. * (**) (***) indicates significance at the 10 (5) (1) percent level.

4.3 Comparing software development productivity gaps with GDP per capita

In this subsection, we compare our estimated productivities with conventional measures of economic development. Since we rely on city-level data and GDP per capita data at this level of granularity do not exist, we use nighttime luminosity per capita as a proxy for income levels. One problem with nighttime luminosity is that rural or underdeveloped and sparsely populated areas may not emit any light. For this reason, we restrict the analysis to FUAs. In Table 4 we regress our productivity measure on nighttime luminosity per capita. In the first column, we observe a strong positive relationship between our productivity estimates and income levels, proxied by nighttime luminosity per capita, for the sample of all FUAs.

Next, we compare our productivity measure with GDP per capita data from the WDI. As was mentioned above, we need to aggregate our productivity measures at the country level. We use three alternative approaches.

First, we calculate the average productivity in the top 5% of locations within each country. Second, we use population shares of each location within each country and construct population weighted aggregate productivity at the country level. Third, we use the GitHub user shares of each location within each country and construct user weighted aggregate productivity at the country level. The results, presented in columns (2)–(4) of Table 4, show that there is a strong positive relationship between GDP per capita and all three productivity measures.

Having established a positive relationship between our estimated productivity measure and various measures of income, we also want to assess whether gaps in software development productivity are different from gaps in GDP per capita between high and low income countries. To this end, we calculate the difference in average log GDP per capita of countries in the top and bottom GDP per capita deciles. We fix the set of countries in both groups and also calculate the difference between the average log of productivity. The difference in GDP per capita is 4.61 log points (see Table 5). The equivalent figures are 4.13 log points for within-country population-weighted productivity, 4.02 log points for the average productivity of top 5% locations, and 4.46 log points for GitHub user-weighted productivity. It is striking, given the different data sources and methodologies, that according to all three approaches the productivity differences are very close to each other and also to the differences in GDP per capita. However, we know from the macro development literature that the agricultural sector is a major contributor to per capita GDP differences between rich and poor countries (Gollin, Parente, and Rogerson, 2002). Productivity differences in other sectors are smaller. Thus, we want to compare our estimated productivity gaps with non-agricultural sectors. We use data from the WDI on value added and employment in the industry and services sectors to construct productivity gaps for the same set of countries that we classified as belonging to the top and bottom deciles based on GDP per capita (see Appendix A.2 for the methodology). The productivity gap for industry is 3.71 and for services 3.73, which are smaller than our estimated IT productivity

Table 5: Productivity gaps between rich and poor countries

Variables	Productivity gap
GDP per capita	4.61
Industry VA per worker	3.71
Services VA per worker	3.73
IT productivity, top 5%	4.02
IT productivity, population weighted	4.13
IT productivity, user weighted	4.46

Notes: This table present log productivity differences between top and bottom 10% of countries sorted by GDP per capita. The sample is restricted to those countries with multiple locations to reduce the influence of outliers, however the results are robust to using all countries. Productivity gaps are calculated as $log(\bar{X}_{top10}) - log(\bar{X}_{bot10})$, where $\bar{X}$ is the average of the variable shown in the rows of this table in top or bottom income group. Data for GDP per capita, sectoral value added and employment were obtained from WDI. IT productivities are aggregated at the country level by using three approaches: first, by averaging productivity in top 5% locations; second, by applying population weights in each location; third, by applying GitHub user weights in each location.

gaps. This means that in terms of productivity in the software development sector, poor countries perform slightly worse than they do in other non-agricultural sectors.

4.4 Trade in ideas

In Section 2 under **Forks and pull requests** we discussed that our data allow us to study trade of ideas and final software utilizing forks. In this case the analogue of equation 1, which formalized the construction of the flow of code, is given by:

$$\tilde{X}_{ij} = \sum_{k \in K} fork_{ik} \times 1[owner_{jk} = 1], \tag{6}$$

where $\tilde{X}_{ij}$ is the flow of final software from city j to city i, $fork_{ik}$ is the number of forks on project k by other projects with owners from city j and $1[.]$ is the indicator function equal to 1 if the owner of project k is located in city j. Note that for the construction of this measure we only use the second category of forks we described in the data section, as those capture the dimension of trade in ideas.

Table 6: Trade in ideas

	(1)	(2)	(3)	(4)
	$\tilde{X}_{ij}/\tilde{X}_{ii}$	Comparative advantage in ideas over services		
Log distance in miles	-0.4376***			
	(0.0069)			
Log GDP per capita		0.8216***	0.3325**	0.1601
		(0.1904)	(0.1359)	(0.1108)
Controls	Yes	No	No	No
Sample	FUA + Admin	Country level	Country level	Country level
Aggregation method		Average of top 5%	Population weighted	User weighted
Observations	11,811,410	119	119	119
R-squared	0.6628	0.1291	0.0546	0.0223
F		18.62	5.988	2.088

Notes: In column (1), the dependent variable is log productivity for trade in ideas estimated from the model. Controls include binary dummies for the same country, shared borders and shared official languages. In columns (2) - (4), the dependent variable is the ratio of productivities for trade in ideas and trade in services aggregated to the country level using three different approaches: first, by averaging the ratio in top 5% locations (column 2); second, by applying population weights in each location (column 3); third, by applying GitHub user weights in each location (column 4). For the country level regressions, we restrict the sample to those countries with multiple locations to reduce the influence of outliers. The results are robust to using all countries, and the estimate in column (4) becomes statistically significant. Standard errors are robust. * (**) (***) indicates significance at the 10 (5) (1) percent level.

Going back to the model described in Section 3.1, we now assume that the unit of production is a project owner located in city i who produces a differentiated software q. On the demand side other project owners decide from which project to fork. We follow the same steps as above to estimate a gravity equation and obtain measures of productivity of final software generation. Column 1 of Table 6 present the results of the distance elasticity for trade in ideas/final software. The estimated coefficient is smaller compared to trade in software code, which suggests that ideas flow more freely in space, yet not fully void of frictions.

Final product ownership generates more value than coding, which is why developers individually and software production locations collectively strive to move up in the value chain and provide successful final software products (Arora, Arunachalam, Asundi, and Fernandes, 2001). To better understand the positions of different geographic locations in the value chain, we construct a measure of comparative advantage for idea production versus provision of coding services. We back out productivities in idea production equivalently to the approach for software development services that were presented in Table 3, however using the flow data based on equation 6. Then we construct the ratio of productivity in ideas over productivity in services at the location level and aggregate it to the country level following the previous three aggregation approaches. We regress the resulting ratio on GDP per capita. The results of this exercise are presented in columns (2)-(4) of Table 6. For all three aggregation approaches we observe a positive relationship between GDP per capita and comparative advantage in idea production, while in two cases the estimated coefficients are statistically significant. These results suggest that higher income countries have a comparative advantage in idea production compared to coding services.

5 Migration and Sorting

In this section, we study the migration of human capital across and within countries. We begin from the individual perspective, asking whether developers selectively migrate based on their productivity and how migrants perform in their destinations relative to the existing local developer pool. We then move to a country-level perspective and assess whether the individual migration patterns we observe amount to aggregate brain drain. Finally, we decompose the extent to which productivity differences across cities are driven by location-specific factors versus individual ability.

5.1 Selective individual migration

We are particularly interested in determining whether there is quality-based selection into locations. To assess this, we construct an individual-level migration variable, which requires that we observe individuals in both our 2019 and 2021 snapshot of the data and that they report their location in both years.[14] The resulting sample comprises about 1.67 million users,

[14] We apply the same data cleaning efforts to the 2019 snapshot of the data that we described in Section 2 for the 2021 snapshot of the data.

of whom about 85,300 migrate, 32,300 between countries and 53,000 within countries. At the country level, the largest gross outflows of migrants are from the US, India, the UK, China, Brazil and Canada, while countries with largest gross inflows are the US, Germany, the UK, Canada and the Netherlands. Figure F4 illustrates some of the largest bilateral migration flows.

We combine this information about migration decisions with the individual-level quality scores that were constructed as an intermediate step to assemble the city ranking according to *Approach 2*. We regress a dummy that indicates whether an individual migrated or not on this measure. The results are presented in panel A, columns (1) – (3) of Table 7. We observe a positive and statistically highly significant coefficient that is robust to different fixed effect structures – the most rigorous of which includes destination country and origin city fixed effects. In this case, migrants from the same city of differing quality lend the identifying variation. In panel B we assign individuals to quartiles based on their score and estimate the same specifications by using indicator variables for each quartile. We observe that the estimated coefficients increase monotonically in all specification. In columns (4) and (5) of the table, we study differences in within and across country migration in relation to our measure. The observed effects are similar for both types.

To address the question of quality-based sorting, we construct an indicator for upward and downward migration. The indicator is equal to 1 if the destination city of the migrant is ranked higher than the origin city based on the estimated productivities from the model. The results for the continuous score and quartiles dummies are presented in columns (1) to (3) of Table 8. In both panel A and B, we observe that the coefficient on upward migration is larger than that on downward migration. The fact that the coefficient on downward migration is positive is not unexpected, because we know from the literature that higher-skilled individuals are more mobile (Borjas, Bronars, and Trejo, 1992). In column (3) of Table 8, we restrict the sample to migrants only, thereby eliminating potential confounding effects arising from selection into migration, and still observe a positive significant coef-

Table 7: Individual quality and likelihood to migrate

	(1) Migrated	(2) Migrated	(3) Migrated	(4) Migrated within country	(5) Migrated across country
Panel A:					
Log individual score	0.2242***	0.1894***	0.2110***	0.2125***	0.2040***
	(0.0099)	(0.0070)	(0.0053)	(0.0068)	(0.0155)
Observations	998,449	997,821	991,634	970,699	963,705
Pseudo R2	0.0162	0.0731	0.0969	0.0803	0.239
Panel B:					
2nd quartile	0.6819***	0.6430***	0.6814***	0.7521***	0.5295***
	(0.0189)	(0.0249)	(0.0244)	(0.0191)	(0.0477)
3rd quartile	1.0236***	0.9522***	0.9786***	1.0427***	0.8335***
	(0.0189)	(0.0275)	(0.0278)	(0.0198)	(0.0562)
4th quartile	1.4699***	1.3201***	1.3428***	1.4477***	1.1211***
	(0.0161)	(0.0374)	(0.0374)	(0.0205)	(0.0855)
Observations	1,670,503	1,669,538	1,658,897	1,627,531	1,610,751
Pseudo R2	0.0474	0.104	0.124	0.100	0.263
Origin country FE	X	X			
Destination country FE		X	X		X
Origin city FE			X	X	X
Number migrants	85,289	85,289	85,289	53,036	32,253

Notes: In columns (1) - (3) the dependent variable is an indicator variable that is equal to one if an individual's location changed comparing the 2019 and 2021 snapshots of the GitHub database. In column (4) we consider location changes within the same country only, and in column (5) changes to locations in another country only. The individual quality score is based on the centrality of the individual in the follower network. Panel A presents results for the log of this individual score, whereas in panel B we construct dummies for the quality score quartile an individual belongs to. All specifications are estimated by PPML. The fixed effects employed in each regression are marked in the table. Standard errors are clustered at the level of origin cities. * (**) (***) indicates significance at the 10 (5) (1) percent level.

ficient. This restriction also addresses concerns that reporting a change in location may be correlated with the quality measure of software developers. The results of this table indicate that (i) higher quality software developers are more likely to migrate in general; (ii) among migrants, those of higher quality are more likely to migrate to better locations and those of lower quality to worse locations.

While we demonstrated that our measure of a location's productivity is well correlated with income levels, it might be the case that individu-

Table 8: Directional migration of individuals based on individual quality

	(1) Up migration (quality)	(2) Down migration (quality)	(3) Up migration (quality)	(4) Up migration (GDP pc)	(5) Down migration (GDP pc)	(6) Up migration (GDP pc)
Panel A:						
Log individual score	0.2277***	0.1772***	0.0252***	0.3361***	0.2392***	0.0152***
	(0.0062)	(0.0068)	(0.0033)	(0.0092)	(0.0107)	(0.0034)
Observations	969,506	936,543	67,431	900,986	843,212	25,381
Pseudo R2	0.183	0.112	0.111	0.108	0.0957	0.117
Panel B:						
2nd quartile	0.6817***	0.6414***	0.0038	0.5757***	0.7993***	-0.0084
	(0.0245)	(0.0375)	(0.0112)	(0.0354)	(0.0377)	(0.0121)
3rd quartile	0.9822***	0.9251***	0.0482***	1.0045***	1.1334***	0.0020
	(0.0234)	(0.0445)	(0.0095)	(0.0321)	(0.0363)	(0.0087)
4th quartile	1.3684***	1.2275***	0.0791***	1.5501***	1.5464***	0.0270***
	(0.0299)	(0.0558)	(0.0096)	(0.0256)	(0.0412)	(0.0080)
Observations	1,624,772	1,564,361	81,621	1,496,137	1,408,209	30,376
Pseudo R2	0.199	0.132	0.110	0.123	0.113	0.116
Destination country FE	X	X	X			
Origin city FE	X	X	X	X	X	X
Sample	All	All	Migrants	All	All	Migrants
Number migrants	48,793	34,354	48,793	21,564	10,689	21,564

Notes: The dependent variable *up migration* (*down migration*) is an indicator variable that is equal to one if an individual migrated to a location that is more (less) productive than their previous location in columns (1) to (3), or to a country with higher (lower) GDP per capita than their previous location in columns (4) to (6). In columns (3) and (6) we restrict the sample to migrants only. The individual quality score is based on the centrality of the individual in the follower network. Panel A presents results for the log of this individual score, whereas in panel B we construct dummies for the quality score quartile an individual belongs to. All specifications are estimated by PPML. The fixed effects employed in each regression are marked in the table. Standard errors are clustered at the level of origin cities. * (**) (***) indicates significance at the 10 (5) (1) percent level.

als choose to migrate to a lower quality location with higher income levels. To investigate this, we regress our individual level quality scores on a dummy indicating an upward or downward migration based on the origin and destination countries' relative GDP per capita. The results are presented in columns (4) to (6) of Table 8 and are similar to the ones based on locations' productivities. We observe that individuals with higher quality scores are more likely to migrate in both directions, but the coefficient on upward migration is higher. In column (6) we again restrict the sample to cross-country migrants to remove systematic differences between migrants and non-migrants, as well as within-country migrants and cross-country migrants. The results show that among migrants, the higher-skilled ones are more likely to move up.

5.2 Migrants in their destinations

Next we assess migrants' relative quality compared to the quality of residents in their destination location before migrating. To this end we construct a dummy variable that indicates whether an individual is above or below the median quality of GitHub users in their destination city. In panel A column (1) of Table 9 we regress the migration dummy on this measure, employing destination city fixed effects. By design the outcome has a sample mean close to 0.5, such that a positive coefficient in this regression indicates that migrants are on average better than the median user in their destination. Vice versa, a negative coefficient would suggest the opposite. The estimated effect implies that an average migrant is better than the median of users in 76% of cases in our sample.[15] In columns (2) and (3) we decompose migration into upward and downward migration based on locations' productivities as in Table 8. The results show that on average this finding holds even in the case of an upward migration move. Naturally, the estimated coefficient is larger for downward migration moves, as the median quality of software developers is lower in these cases. In columns (4) and (5) we replicate the specification but for upward and downward migration defined by GDP per capita differences. The general patterns and estimated coefficients turn out to be very similar to the productivity based results.

In panel B of Table 9 we investigate how migration decisions affect the migrants' individual position in the quality score distribution. We calculate the change in quality score quartile based on the distribution of quality scores in origin and destination location in 2019, that is prior to migration taking place. We regress the change in quartile on the different migration dummies we have employed in panel A. The results are consistent with the evidence we compiled so far. Migrants move on average down the quality score distribution, which is driven by moves to more productive and

[15] We transform the semi-elasticity of 0.4219 according to the following formula: $(100 * (exp(\beta) - 1))$. Multiplying the baseline likelihood of 0.5 with the resulting 52.49% yields around 26% higher likelihood of being above the median quality in the destination.

Table 9: Migrants comparative quality in the destinations

	(1) Above median score in destination	(2) Above median score in destination	(3) Above median score in destination	(4) Above median score in destination	(5) Above median score in destination
Panel A:					
Migrated	0.4219***				
	(0.0087)				
Up migration (productivity)		0.3811***			
		(0.0053)			
Down migration (productivity)			0.4540***		
			(0.0141)		
Up migration (GDP per capita)				0.3644***	
				(0.0160)	
Down migration (GDP per capita)					0.4377***
					(0.0128)
Observations	1,669,111	1,663,694	1,663,694	1,669,111	1,669,111
Pseudo R2	0.00519	0.00345	0.00348	0.00267	0.00252
	(1) Δ quartile individual score	(2) Δ quartile individual score	(3) Δ quartile individual score	(4) Δ quartile individual score	(5) Δ quartile individual score
Panel B:					
Migrated	-0.0397***				
	(0.0133)				
Up migration (productivity)		-0.1036***			
		(0.0066)			
Down migration (productivity)			0.0515**		
			(0.0227)		
Up migration (GDP per capita)				-0.1254***	
				(0.0203)	
Down migration (GDP per capita)					0.0298*
					(0.0180)
Observations	1,670,319	1,663,732	1,663,732	1,670,319	1,670,319
R-squared	0.2541	0.0474	0.0177	0.2642	0.2483
Destination city FE	X	X	X	X	X
Number migrants	85,289	48,793	34,354	21,564	10,689

Notes: The dependent variable in panel A is an indicator variable that is equal to one if an individual has a higher quality score than the average user in the destination location. In panel B the dependent variable is the difference of individuals' quality score quartiles between their location in 2019 and their location in 2021, calculated according to the distribution of quality scores in 2019 in both locations. Explanatory variables are: *Migration* - a dummy for migration; *Up migration* a dummy if migration takes place to a location with higher productivity or to a country with higher GDP per capita; *Down migration* a dummy if migration takes place to a location with lower productivity or a country with lower GDP per capita. The individual quality score is based on the centrality of the individual in the follower network. All specifications in panel A are estimated by PPML, in panel B by OLS. The fixed effects employed in each regression are marked in the table. Standard errors are clustered at the level of origin cities. * (**) (***) indicates significance at the 10 (5) (1) percent level.

higher income locations. Moves to less productive places see the migrant on average move up the quality score distribution.

5.3 Aggregate flows of migration

In the previous subsection we documented strong sorting patterns using individual level migration decisions. These patterns imply that locations and countries with an initially low stock of individuals with high quality are losing their best experts. In the literature this phenomenon is referred to as brain drain. In this subsection we investigate whether the migration pattern at the individual level has tractable implications at the aggregate level. To this end, we construct three measures: net migration flows, gross inflows and gross outflows.

We aggregate the individual quality scores at the country level in 2019 to calculate the initial stock of human capital. We then construct our measure of gross inflow, as the sum of scores of individuals who migrated to a country in 2021. Equivalently, we calculate the measure of gross outflow as the sum of scores of migrants leaving the country. We divide both the inflow and the outflow measure by the initial stock of human capital we calculated for 2019, to express them in relative terms. Net migration is constructed as the ratio of the stock of human capital in 2021, over the initial stock in 2019. In Table 10 we regress these measures on GDP per capita. To reduce the noise in this regression, we drop countries that have less than 20 users in 2019 in panel A. In panel B we increase the threshold to at least 150 users.

The results show that countries with higher GDP per capita experience positive net migration. This is driven by larger inflows, indicated by the positive coefficients in both panels in the third column, which are larger than the coefficients for outflows in the second column. This resembles a setting in which software developers from high-income countries might migrate to other high income countries, and software developers from low-income countries tend to migrate strictly upwards. The results confirm our conjecture based on the individual level regressions that wealthier countries are attracting talent, while poorer countries are losing talent.

5.4 Individuals vs. locations

We document large differences in average productivity across locations. An important policy question is what drives these gaps. Do some cities possess higher fundamental productivities, are there agglomeration effects, or

Table 10: Migration flows at the country level

	(1) Net migration	(2) Out-migration	(3) In-migration
Panel A:			
Log GDP per capita	0.0270***	0.0038	0.0232**
	(0.0115)	(0.0030)	(0.0097)
Observations	131	131	131
R-squared	0.0732	0.0069	0.0596
Panel B:			
Log GDP per capita	0.0329***	-0.0057	0.0272***
	(0.0080)	(0.0035)	(0.0083)
Observations	119	119	119
R-squared	0.1270	0.0132	0.1033

Notes: In Panel A we require countries to have any in- and out-migration flows and more than 20 GitHub users. In Panel B we increase this threshold to more than 150 GitHub users. Standard errors are robust. * (**) (***) indicates significance at the 10 (5) (1) percent level.

do more productive individuals sort into places with superior amenities?[16] Our ability to separate these channels is limited by data availability - specifically, the absence of follower-network information in the 2021 data snapshot. Nevertheless, we can make progress by adapting the literature that uses worker-firm matched data to disentangle worker and firm productivities (Abowd, Kramarz, and Margolis, 1999) to our setting. Since we do not have full information on firms for which individuals work, we replace firms with cities and exploit individual migration across cities for identification. In doing so we follow the recent macroeconomic literature which uses data on workers' movements from one country to another or data on firms operating in multiple countries (Martellini et al., 2024; Alviarez, Cravino, and Ramondo, 2023). We estimate the following regression equation

[16]Many West Coast US cities, which rank highly in our measure, also feature mild weather and other natural amenities.

$$Productivity_{it} = \mu_i + \psi_{j(i,t)} + \tau_t + \epsilon_{it}, \quad (7)$$

where $Productivity_{it}$ is the productivity of individual i in period t, μ_i is an individual fixed effect (intrinsic ability), $\psi_{j(i,t)}$ is the productivity of the city j where i resides in period t, τ_t is a time fixed effect and ϵ_{it} is the error term. The variance decomposition of equation 7 provides a simple accounting of how much of the overall dispersion reflects individual ability versus location characteristics.

To implement (7), we require a time-varying productivity measure at the individual level. Because the 2021 snapshot lacks follower data, we use an earlier snapshot from 2018 to construct a follower-based measure analogous to the measure based on the 2019 data snapshot. This approach has a drawback: Since GitHub's user base was considerably smaller in 2018, we are left with far fewer migrants – only about 9% as many as in the 2019-2021 period – and therefore less identifying variation.

Estimating (7) on the 2018-2019 panel and decomposing the variance of $Productivity_{it}$ reveals that the individual fixed effect accounts for more than 99% of the variance, while the city fixed effect contributes negligibly. This suggests that individual ability dominates location in explaining productivity differences. This finding should however not be read as implying that policy is irrelevant. Amenities and broader place-based factors, which are shaped by policy, may play a crucial role by attracting and retaining high-ability developers even if, conditional on sorting, city fixed effects contribute little additional explanatory power.

There are caveats to the Abowd et al. (1999) approach we take. The decomposition does not allow us to disentangle fundamental city productivity from agglomeration effects. However, given the near-zero contribution of the combined city effect, this distinction is relatively unimportant for our context. Further, we acknowledge that the quantitative result reported in this section should be interpreted with caution because of the reduced sample size due to the data constraints discussed above.[17] In addition, the short interval between data snapshots prevents us from capturing agglomeration effects that materialize only over longer horizons, particularly positive network effects that may grow nonlinearly over time. Finally, the literature has shown that the Abowd et al. (1999) approach can yield biased estimates when worker mobility is limited (Bonhomme, Holzheu, Lamadon, Manresa, Mogstad, and Setzler, 2023).

6 Conclusions

In this paper we bring new empirical evidence to the debate on the role high-skilled tradable services play in economies around the world, and for the development process of low-income countries.

We study the software development industry, specifically the large and commercially important sector of open source development, by utilizing detailed data at the level of individual software developer. Our main contribution is the estimation of productivity levels in 5,200 locations around the world. Our results show that there are large differences in productivity levels within and across countries, which are indicative of human capital differences across space. We find that the productivity gaps between the richest and poorest countries in software development are somewhat larger than for the broadly defined manufacturing and services sectors. Developing countries are seemingly not able to leverage the fact that transportation costs are near zero to generate exports, likely because of information frictions that are captured in the sizable distance elasticities we measure. Moreover, we find evidence of "brain drain" – that is, a sorting pattern in which the best software developers from less developed countries or cities with low levels of productivity move to more productive locations. This

[17] As a robustness check, we replicate the decomposition using commit counts in 2019-2021 and before 2019 as an alternative measure of productivity. This approach preserves a larger sample, but raw commit counts are an imperfect proxy for individual productivity. Nevertheless, the results from this robustness exercise are qualitatively in line with those of the main decomposition.

exacerbates existing differences.

These findings present a rather bleak picture for many low-income countries. Yet there are notable bright spots such as Bengaluru and Ho Chi Minh City, which have very high productivity levels and are ranked among the global leaders. Understanding the evolution of the ICT sector in these places can offer practical lessons for other locations in developing countries seeking to raise productivity in this sector.

In addition to identifying these model cases that can inform other developing countries, our analysis yields two concrete policy implications. First, because distance appears to matter less for lower-level programming languages, targeted skill development in these areas could help geographically isolated places integrate into global markets. However, this path is not costless since such languages typically have steeper learning curves. Second, we find strong selection patterns that shape city-level productivity. We believe that amenities play an important role in migration decisions. Both national and local governments have levers here: investing in quality-of-life infrastructure and reducing frictions to mobility can attract and retain the talent that makes up the high-productivity ICT clusters.

There are a number of important questions that require further attention. For example, our results suggest little role for agglomeration. However, our analysis is based on a short interval and one cannot rule out that such forces operate over longer time horizons. Moreover, our decomposition exercise is based on the AKM approach and we lack a sound identification strategy. We believe that as more data becomes available to researchers it will be possible to tackle this issue more credibly by using longer time horizons and better identification strategies based on plausibly exogenous shocks.

References

Abowd, J. M., F. Kramarz, and D. N. Margolis (1999): "High Wage Workers and High Wage Firms," *Econometrica*, 67, 251–333.

Akcigit, U., S. Baslandze, and S. Stantcheva (2016): "Taxation and the International Mobility of Inventors," *American Economic Review*, 106, 2930–81.

Allen, T. (2014): "Information Frictions in Trade," *Econometrica*, 82, 2041–2083.

Allen, T. and C. Arkolakis (2018): "Modern Spatial Economics: A Primer," in *World Trade Evolution*, ed. by L. Y. Ing and M. Yu, Routledge.

Alviarez, V., J. Cravino, and N. Ramondo (2023): "Firm-Embedded Productivity and Cross-Country Income Differences," *Journal of Political Economy*, 131, 2289–2327.

Arora, A., V. Arunachalam, J. Asundi, and R. Fernandes (2001): "The Indian software services industry," *Research Policy*, 30, 1267–1287.

Bai, J., M. X. Chen, J. Liu, X. Mu, and D. Y. Xu (2022): "Stand Out from the Millions: Market Congestion and Information Friction on Global E-Commerce Platforms," .

Bias, B. and S. Ma (2023): "The Education-Innovation Gap," Working paper.

Bonhomme, S., K. Holzheu, T. Lamadon, E. Manresa, M. Mogstad, and B. Setzler (2023): "How Much Should We Trust Estimates of Firm Effects and Worker Sorting?" *Journal of Labor Economics*, 41, 291–322.

Bordeu, O. (2023): "Commuting infrastructure in fragmented cities," *Job Market Paper, University of Chicago Booth School of Business*, 6, 64.

BORJAS, G. J., S. G. BRONARS, AND S. J. TREJO (1992): "Self-selection and internal migration in the United States," *Journal of Urban Economics*, 32, 159–185.

BRIN, S. AND L. PAGE (1998): "The anatomy of a large-scale hypertextual Web search engine," *Computer Networks and ISDN Systems*, 30, 107–117, proceedings of the Seventh International World Wide Web Conference.

BUERA, F. J. AND J. P. KABOSKI (2012): "The Rise of the Service Economy," *American Economic Review*, 102, 2540–69.

CLEMENS, M. A. (2013): "Why Do Programmers Earn More in Houston Than Hyderabad? Evidence from Randomized Processing of US Visas," *American Economic Review*, 103, 198–202.

EATON, J. AND S. KORTUM (2002): "Technology, Geography, and Trade," *Econometrica*, 70, 1741–1779.

——— (2018): "Trade in Goods and Trade in Services," in *World Trade Evolution*, ed. by L. Y. Ing and M. Yu, Routledge.

ECKERT, F. (2019): "Growing Apart: Tradable Services and the Fragmentation of the U.S. Economy," Working paper.

FACKLER, T., M. HOFMANN, AND N. LAURENTSYEVA (2023): "Defying Gravity: What Drives Productivity in Remote Teams?" Tech. rep., Discussion Paper.

FALLY, T. (2015): "Structural gravity and fixed effects," *Journal of International Economics*, 97, 76–85.

FREIRE, S., K. MACMANUS, M. PESARESI, E. DOXSEY-WHITFIELD, AND J. MILLS (2016): "Development of new open and free multi-temporal global population grids at 250 m resolution," *Population*, 250.

GARICANO, L. (2000): "Hierarchies and the Organization of Knowledge in Production," *Journal of Political Economy*, 108, 874–904.

GERVAIS, A. AND J. B. JENSEN (2019): "The tradability of services: Geographic concentration and trade costs," *Journal of International Economics*, 118, 331–350.

GOLDBECK, M. (2023): "Bit by bit: colocation and the death of distance in software developer networks," Tech. rep., Discussion Paper.

GOLLIN, D., S. PARENTE, AND R. ROGERSON (2002): "The Role of Agriculture in Development," *American Economic Review*, 92, 160–164.

GOUSIOS, G. (2013): "The GHTorent dataset and tool suite," in *2013 10th Working Conference on Mining Software Repositories (MSR)*, IEEE, 233–236.

HALL, R. E. AND C. I. JONES (1999): "Why do Some Countries Produce So Much More Output Per Worker than Others?*," *The Quarterly Journal of Economics*, 114, 83–116.

HENDRICKS, L. AND T. SCHOELLMAN (2018): "Human capital and development accounting: New evidence from wage gains at migration," *The Quarterly Journal of Economics*, 133, 665–700.

HSIEH, C.-T. AND E. ROSSI-HANSBERG (2023): "The Industrial Revolution in Services," *Journal of Political Economy Macroeconomics*, 1, 3–42.

KLENOW, P. J. AND A. RODRÍGUEZ-CLARE (1997): "The Neoclassical Revival in Growth Economics: Has It Gone Too Far?" *NBER Macroeconomics Annual*, 12, 73–103.

LEVCHENKO, A. A. AND J. ZHANG (2016): "The evolution of comparative advantage: Measurement and welfare implications," *Journal of Monetary Economics*, 78, 96–111.

MARTELLINI, P., T. SCHOELLMAN, AND J. SOCKIN (2024): "The Global Distribution of College Graduate Quality," *Journal of Political Economy*, 0, 000–000.

MORENO-MONROY, A. I., M. SCHIAVINA, AND P. VENERI (2021): "Metropolitan areas in the world. Delineation and population trends," *Journal of Urban Economics*, 125, 103242.

RUGGLES, S., S. FLOOD, R. GOEKEN, M. SCHOUWEILER, AND M. SOBEK (2022): "Integrated Public Use Microdata Series: Version 12.0 [dataset]," Tech. rep., Minneapolis, MN: IPUMS.

STARTZ, M. (2016): "The value of face-to-face: Search and contracting problems in Nigerian trade," *Available at SSRN 3096685*.

WACHS, J. (2023): "Digital traces of brain drain: developers during the Russian invasion of Ukraine," *EPJ Data Science*, 12, 14.

WACHS, J., M. NITECKI, W. SCHUELLER, AND A. POLLERES (2022): "The geography of open source software: evidence from github," *Technological Forecasting and Social Change*, 176, 121478.

WAUGH, M. E. (2010): "International Trade and Income Differences," *American Economic Review*, 100, 2093–2124.

WRIGHT, N. L., F. NAGLE, AND S. GREENSTEIN (2023): "Open source software and global entrepreneurship," *Research Policy*, 52, 104846.

Appendix

A Additional data description

A.1 Spatial data

We employ a number of supplementary data sources, which we combine with our main data by spatial proximity.

Locations We use shape files from the Global Human Settlements Functional Urban Areas dataset, which identifies metropolitan areas and their surrounding commuting zones around the world. The methodology of creating these functional urban areas (FUAs) is laid out in Moreno-Monroy, Schiavina, and Veneri (2021).[18] We map GitHub users based on their geocoordinates to the FUAs. To capture less densely populated areas as well, we then group together users that fall outside the borders of FUAs and assign them to the admin-2 region they are located in. Shapefiles for administrative borders come from the Database of Global Administrative Areas (GADM). In the paper we use the terms locations and cities interchangeably. We drop locations with 10 or less unique users to avoid calculating very noisy aggregate measures at the location level. We then compute the distance between two locations as the geodesic distance between the mean coordinates (latitude and longitude) of all users in each location using Python's `GeoPy` function. The top 20 locations in terms of the number of users are displayed in Table A4. We arrive at a final sample of 5,268 locations in 190 countries. We map all our other data sources into these geographic areas; Figure F2 provides a visual example of this approach for nighttime luminosity, GitHub users and FUAs.

Population We extract population numbers for the locations we consider from the Global Human Settlements population grid, which is a spatial

[18] For some countries alternative definitions of urban areas are available – for example, the Metropolitan Statistical Areas or Commuting Zones for the US – but such maps are not available for all countries and approaches may differ across countries.

raster that depicts the distribution of the residential population. We utilize the grid at a resolution of 1 kilometer; each cell has a value for the predicted number of people living in that area. The construction of the raster is explained in Freire, MacManus, Pesaresi, Doxsey-Whitfield, and Mills (2016). We overlay that raster with the FUA and admin-2 borders shape files to extract the sum of population at our level of observation.

Nightlights We obtain nighttime luminosity by overlaying a spatial raster of nighttime luminosity provided by the Earth Observation Group with our FUA and admin-2 border shape files. We utilize the V2.1 annual version of VIIRS to extract the average sum of nocturnal light omitted at the location level. This version of nighttime data has the advantage that it is not top coded, making cross-country comparisons of cities with potentially strongly diverging luminosity levels more precise.

US air traffic We combine two data sources: administrative records on U.S. domestic air traffic from the U.S. Bureau of Transportation Statistics (BTS) and an open global database of airport locations from OpenFlights. BTS T-100 Domestic Market Data report monthly passenger and freight traffic for domestic origin–destination markets operated by U.S. air carriers. Each observation corresponds to an origin–destination pair of airports for a given carrier, month, and service class. Data are available on a monthly basis from 1990 onward and we aggregate them to the origin-destination airport-pair level over two periods. First, from January 2009 to April 2021, capturing the period from the inception of GitHub to our latest data snapshot. Second, from 1990 to 2008 to capture the period before GitHub existed.

To obtain the geographic location of each airport, we merge the T-100 data by three-letter IATA airport code with the OpenFlights Airports Database which contains latitudes and longitudes. Airports with unmatched codes - typically very small, obsolete, or non-civil airports - are dropped from the analysis. We then map the airports into FUAs and Admin-2 regions based on the geocoordinates.

A.2 Income data

We are interested in relating the differences we measure in human capital across space to income differences. We do so at the level of FUAs for the United States, and globally at the country level.

American Community Survey (ACS) We use the ACS data provided by Ruggles, Flood, Goeken, Schouweiler, and Sobek (2022) to construct wages at the level of Public Use Microdata Areas (PUMAs), which are the smallest identifiable geographic unit in that dataset. They are non-overlapping statistical areas containing no fewer than 100,000 people each. Given that FUAs do not exactly align with PUMAs, we intersect them, and re-weight the average wages thus obtained. We calculate the weights as follows:

$$Weight_{p,F} = \frac{Share\ intersected\ area_{p,F} * Population_p}{Population_{P,F}}, \tag{8}$$

where the index p depicts the individual PUMA, F the FUA it is intersecting with, and P, F all PUMAs intersecting with the same FUA. Figure F3 in the Appendix visualizes the intersection of PUMAs and FUAs.

We use occupational information to identify individuals who are employed in software-related occupations. We identify 14 such occupations, which are listed in Table A3. We have also extended the list by including a broader list of occupations that may require software development skills, such as economist and physicist. This extended list yielded similar results. However, we believe a stricter definition is more appropriate because the fraction of economists engaged in software development is unlikely to be high and this is not their main activity.

Software developer wages We are not aware of any global administrative database on the earnings of software developers. For this reason we utilize data from a survey conducted by *Stack Overflow*, which is a question-and-answer website for programmers and has over 20 million registered users. Every year *Stack Overflow* conducts a survey among its users on various issues related to their professional activity including their salaries. We use the *2023 Developer Survey* since it has broader coverage compared to previous years. Ninety thousand developers from 87 countries responded to the survey. We drop survey responses from users who stated something other than being a software developer by profession or programmer as part of their work, in order to focus on the earnings of IT professionals. Of this sub-sample the number of respondents with non-missing wage income responses ranges from 16,409 in the US to 12 in Senegal, Kuwait and Bahrain. The country with the median number of observations has 135 respondents. We winzorize the wages at the 99% level to reduce the impact of outliers, in particular in the small sample countries. Clearly, this survey comes with limitations but we believe that a comparison of our estimated productivity measure with wages from a survey from a different source is a useful exercise that can potentially support the validity of our estimates.

WDI We obtain GDP per capita in constant 2015 US dollars for the years 2019 and 2021 at the country level. We also obtain data to calculate value added per worker for the industry and services sectors. In particular, we obtain the value added per capita (per worker) by dividing value added in constant 2015 US$ for the respective sector by the number of workers in this sector, which we derive by multiplying the industry employment share in total employment with the product of the employment to 15+ population ratio and the population total age 15-64. We merge this information to our remaining data by 3-letter country codes.

A.3 Representativeness

In the following paragraphs, we provide a more detailed discussion of the representativeness of our sample, given that we are able to map only a subsample of users accurately into locations. We refer to information provided in Section 2, which introduces the users and commits data, along with the individual quality scores generated through *Approach 2* outlined in Section 3.2.

We require the information of users location to attribute commits, which form the basis of the trade flows we construct, to locations. Our dataset comprises 222,167,691 commits from users whose locations were accurately identified following our data cleaning procedures. Additionally, we identify 377,545,627 commits from users without location information. While this constitutes a share of 37%, it is noteworthy that users with location information are far more active; They average 82.6 commits compared to 12.1 commits for users lacking location details. To address the potential skew in commit volume caused by less meaningful commits from users with incomplete profiles, we compute a quality-adjusted share by weighting each commit with the respective user's individual quality score. Consequently, when adjusting for quality scores, we are able to attribute 68.1% of the commit volume to specific locations. Notably, our gravity estimations using raw commit counts and quality adjusted commits deliver similar results (see columns (1) and (5) of Table 1). The fact that there is a large difference in the covered share of commit volume between both approaches, yet the gravity estimation results being close to each other suggests that it is unlikely that there are systematic patterns in terms of not reporting location information.

A.4 Samples

Below in Table A1 we summarize the samples used throughout the empirical analysis. We structure the table into five dimensions of our data: commit-based samples, subnational location and flow samples, country-level samples, individual-level samples, and fork-based samples. For each sample, the table reports the unit of observation, sample size, its role in the paper and where it has been employed. Important to note, in the table we report the total number of observations for each of the samples, which typically differs from the effective number of observations reported in the main text and regression tables.

Table A1: Samples used in the main empirical analyses

Sample	Unit of observation	N	Main use in paper
Panel A: Commit-based samples			
All commits (after dropping bots and self-links)	Commit	599,713,318	Used for coverage analysis (Appendix A.3).
Commits from users without cleaned location	Commit	377,545,627	Used for coverage analysis (Appendix A.3).
Commits from users with cleaned location	Commit	222,167,691	Basis for all location-level flows X_{ij} (equation (1)) and the structural gravity estimation of trade in tasks (Table 1, columns (1)-(4); Table 2; Figure 1; Table E1) and for all downstream city and country productivity measures (Table 3; Table 4; Table 5; Table 6, columns (2)-(4); Figure 2).
Commits from users with cleaned location & follower information	Quality-adjusted Commits		Basis for quality adjusted location-level flows $\hat{X}_{ij}$ (Table 1 col (5)).
Panel B: Subnational (location and flow) samples			
All locations (FUAs + admin-2, locations with >10 users)	Location	5,268 locations in 190 countries	Used to construct bilateral flows X_{ij} and $\tilde{X}_{ij}$, to estimate city productivities, and to map auxiliary data such as population, nightlights, income and flight data.
Functional Urban Areas (FUAs) only	Location (FUA)	2,721	Used where analysis is restricted to metropolitan areas (e.g. regression of productivity on nightlights per capita; Table 4, column (1); US wage validation by FUA in Figure 2a).

Continued on next page

Table A1: Samples used in the main empirical analyses (Continued)

Sample	Unit of observation	N	Main use in paper
Bilateral flows between all locations (FUA + admin-2)	Ordered location pair (i, j)	27,133,681	Baseline gravity sample for trade in tasks with importer and exporter fixed effects (Table 1, columns (1) and (4)); also underlying sample for robustness with quality-weighted flows (Table 1, column (5)) and for specification with language fixed effects.
Bilateral flows between FUAs only (global)	Ordered location pair (i, j)	7,273,809	Gravity estimation restricted to FUAs (Table 1, column (2)).
Bilateral flows between US FUAs only	Ordered location pair (i, j)	66,564	US-only gravity specification for trade in tasks (Table 1, column (3)).
Bilateral flows between US locations (FUA + admin-2) with air-traffic controls	Ordered location pair (i, j)	1,159,929	Gravity regressions augmented with passenger and freight flows (Table 2, columns (1)-(4)).
Language-specific bilateral flows (languages with $\geq$ 1,000,000 commits)	Ordered location-by-language pair (i, j)	Varies by language: min 3,904,576 (*Go*), max 18,983,449 (*JavaScript*)[a]	Used to estimate heterogeneous distance elasticities by programming language and in pooled regression with language fixed effects (Figure 1; Section 4.1.2).
Flows for small-team projects (team size $<$ 50 contributors)	Ordered location pair (i, j)	23,843,689	Gravity estimations by project team size; distance elasticity for small teams (Table E1, column (1)).

Continued on next page

Table A1: Samples used in the main empirical analyses (Continued)

Sample	Unit of observation	N	Main use in paper
Flows for large-team projects (team size $\geq$ 50 contributors)	Ordered location pair (i, j)	1,844,164	Gravity estimations by project team size; distance elasticity for large teams (Table E1, column (2)).
Panel C: Country-level samples			
Countries with at least one GitHub location	Country	190	Coverage of GitHub data across countries.
Countries with productivity measures for multiple locations and income data	Country	131	Country-level regressions relating aggregate IT productivity to GDP per capita and for productivity gap calculations (Table 4, columns (2)–(4); Table 5). Also used in the cross-country wage validation (Figure 2b).
Countries with $\geq$ 20 GitHub users, both in- and out-migration flows and income data	Country	131	Brain-drain regressions relating aggregate migration of developers to GDP per capita (Table 10, Panel A).
Countries with $\geq$ 150 GitHub users, both in- and out-migration flows and income data	Country	119	Stricter brain-drain sample to check robustness to user-count thresholds (Table 10, Panel B).
Countries with multiple locations, trade-in-ideas and trade-in-services productivity measures and income data	Country	119	Comparative advantage in ideas over services regressions based on country-level ratios of ideas to services productivity (Table 6, columns (2)–(4)).

Continued on next page

Table A1: Samples used in the main empirical analyses (Continued)

Sample	Unit of observation	N	Main use in paper
Panel D: Individual-level samples			
All registered GitHub accounts (2021 snapshot)	Users	45,838,859	Overall platform size.
Users with any self-reported location string	User	3,685,428	Starting sample for location cleaning.
Users with cleaned subnational locations (FUA or admin-2)	User	2,748,631	Base sample for all analyses requiring a location: construction of city and country productivities, migration and sorting analyses, and all regressions using location-level variables (Sections 3–5).
Users with follower information (at least one following or follower) from 2019 data snapshot	User	5,375,944	Used to construct individual quality scores via PageRank; used as the key covariate in individual-level and aggregate migration regressions (Sections 5.1–5.3) and the AKM productivity decomposition exercise in Section 5.4. Aggregated to cities for alternative productivity rankings (Table 3, Approach 2), and to universities for alternative validation (Table D1). Used to construct quality weights for $\hat{X}_{ij}$ (Table 1, column (5)).
Migration panel: users observed with cleaned location in both 2019 and 2021 data snapshots	User	1,700,520	Panel of developers for migration analysis; underlying sample for migration decisions and sorting by quality (Section 5).

Continued on next page

Table A1: Samples used in the main empirical analyses (Continued)

Sample	Unit of observation	N	Main use in paper
Migrants in migration panel (any location change between 2019 and 2021)	User	108,276 (66,961 within-country; 41,315 across-country)	Number of movers underlying the intensive-margin migration results, including upward vs downward migration by city productivity and GDP per capita (Tables 7, 8 and 9) as well as the descriptive country-level migration flows (Figure F4).
2018–2019 panel with follower-based individual productivity measures	User	1,017,623	Used to decompose variance of individual vs location fixed effects in productivity (Section 5.4).
University-affiliated users (US, UK, Germany)	User with university affiliation	56,305	Basis for university rankings constructed from aggregated individual scores (Table D1).
Panel E: "Fork"-based samples			
Bilateral flows for trade in ideas (fork-based final software flows)	Ordered location pair (i, j)	27,573,001	Gravity equation for trade in ideas/final software (Section 4.4; Table 6, column (1)).

[a] All small languages (< 1,000,000 commits) combined have 26,112,100 location pairs.

Note: This table summarizes the main samples used in the empirical analysis along five dimensions of our data: commits, subnational locations and flows, countries, individuals and forks.

Table A2: Share of local connections by team size

Team size	Observations	Local share
2-5	296,348	0.590
6-20	167,551	0.492
21-100	87,616	0.403
$>$100	88,764	0.164

Notes: This table shows the average share of local connections across projects of a given size team. A connection is an undirected link between two users.

Table A3: IT occupations

Code	Description
1005	Computer and information research scientists
1006	Computer systems analysts
1007	Information security analysts
1010	Computer programmers
1021	Software developers
1022	Software quality assurance analysts and testers
1031	Web developers
1032	Web and digital interface designers
1050	Computer support specialists
1065	Database administrators and architects
1105	Network and computer systems administrators
1106	Computer network architects
1108	Computer occupations, all other
1240	Other mathematical science occupations

Notes: This table presents the list of occupations in the ACS, which we classify as IT-related. The first column displays occupation codes according to variable *occ*.

Table A4: City user counts

	Location	User count		Location	User count
1	San Jose	106,341	11	Paris	35,273
2	New York	80,284	12	Toronto	33,639
3	London	64,890	13	São Paulo	32,801
4	Bengaluru	62,688	14	Moscow	31,963
5	Beijing	61,580	15	Tokyo	31,525
6	Seattle	46,753	16	Berlin	30,578
7	Los Angeles	44,156	17	Boston	30,405
8	Shanghai	40,411	18	Chicago	28,796
9	Delhi [New Delhi]	38,360	19	Washington D.C.	24,155
10	Guangzhou	35,569	20	Pune	23,316

B The organization of teams

In this section, we study the structure of production teams. Our primary reason for doing so is to understand how to define the flows of software code between locations. However, this touches upon a much broader aspect in the theory of the firm and there is a large literature studying the hierarchies in organizations (Garicano, 2000).

Production teams can be organized in different ways. At one extreme, the production process may be organized in the shape of a star, such that every worker or production unit delivers its output to the central unit. Alternatively, production may be organized as a chain in which each unit delivers its output to the next. Production can also be organized as a fully connected graph in which each individual interacts with everyone else.

We utilize our data to shed light on the structure of software production teams. We construct linkages between individuals based on the follower network within a project. Then, we test whether the owner of the project stands out among others. To that end, we estimate the following specification:

$$y_{ij} = \alpha + \beta_1 Owner_j + \beta_2 Owner_i + \epsilon_{ij}, \tag{9}$$

where y_{ij} is a dummy if individual i follows individual j , $Owner$ is a dummy if the person is the owner of the project and ϵ_{ij} is the error term. If the team is organized as a chain or if everyone interacts with everyone within the network, then the owner should not have a special status and the coefficient $\beta_1 = 0$.

We present the results of our estimations in Table B1. Estimations are conducted for all projects that have more than two participants. In the first column, the only explanatory variable is whether user j is the owner. The estimated coefficient indicates that owners are much more likely to be followed by other project members. Project owners are thus the central figures in projects, and other team members want to be informed about their contributions as well as the issues and pull requests they open (for instance,

Table B1: The structure of collaboration in software production teams

	(1)	(2)	(3)	(4)	(5)	(6)
	i follows *j*	*i* follows *j*	*i* follows *j*	*i* follows *j*	*i* follows *j*	Share of follows
Owner_j	2.0161***	2.1468***	1.5161***	1.3315***	1.3189***	0.9352***
	(0.0014)	(0.0015)	(0.0029)	(0.0035)	(0.0135)	(0.0018)
Owner_i		1.9697***	1.2686***	1.0639***	-6.5629***	
		(0.0016)	(0.0032)	(0.0039)	(0.6575)	
Same country			0.9379***	0.6965***	0.4723***	
			(0.0018)	(0.0025)	(0.0038)	
Same location				0.4274***	0.2323***	
				(0.0025)	(0.0046)	
Team size	> 2	> 2	> 2	> 2	> 100	> 2
Mean	0.015	0.015	0.030	0.031	0.015	0.161
Observations	244,177,260	244,177,260	46,511,340	32,963,226	26,674,020	3,419,080
Pseudo R^2	0.0303	0.0548	0.0528	0.0504	0.0107	0.0323

Notes: Columns (1)-(5) present the estimation results of equation 9, where the dependent variables are dummies taking a value of 1 if contributor *i* follows contributor *j*. Column (6) presents the results of a regression where the dependent variable is the share of follower links of individual *i* among all following links in a given project. All specifications are estimated with PPML. In column (5) the sample is restricted to projects with more than 100 contributors. * (**) (***) indicates significance at the 10 (5) (1) percent level.

specifically those labeled "help wanted" or "good first issue"). In terms of the organizational form of production, this resembles the star mentioned above.

In the second column we include the $Owner_i$ control and find that the estimated coefficient is also sizable. However, the larger coefficient of $Owner_j$ that is statistically significantly different from $Owner_i$ shows that the owner is more likely to be followed than follow others. The average for y_{ij} is 0.015. This indicates that within an average team there are few interactions between a randomly selected pair. By contrast, owners play a central role and maintain bilateral interactions with other contributors.

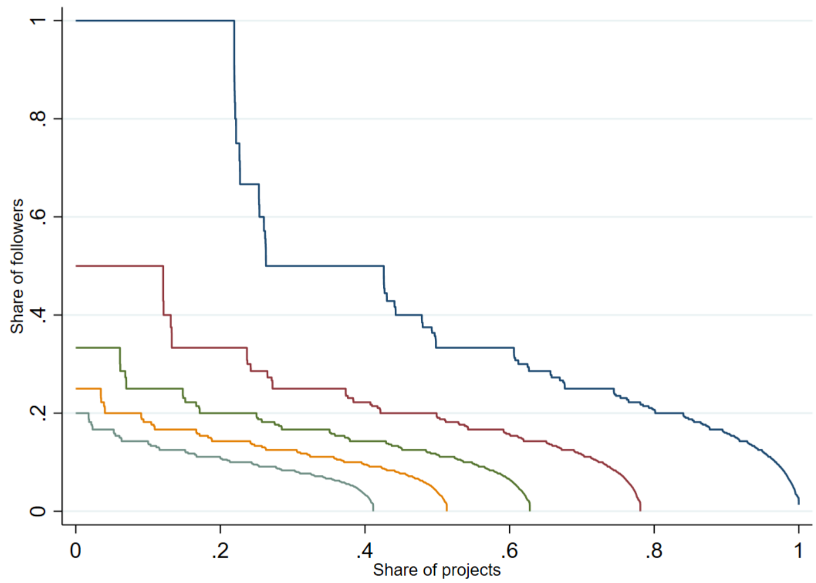
In the following columns we add an indicator variable if a pair of members are located in the same country and city. The estimated coefficients on our variable of interest decrease somewhat but they are still large and statistically significant. In column (5) we report results for the sample of teams with 100 participants or more. The comparison with the results in column (4) reveals that in large projects the role of the owner is as central as in smaller projects. In larger projects the owner is much less likely to follow others, which given the larger team size seems to be intuitive. The distinction between large and small projects is important because in our data such projects contribute disproportionately more to non-local links. More specifically, in teams with 2 to 5 members, links to local members account for 60% of all links, while in teams with more than 100 members such links account for only 15% (see Table A2). In the last column of Table B1, we regress the share of follower links on the owner dummy. Again we obtain a very large and precisely estimated positive coefficient.

In Figure B1, we provide further evidence that within teams a few individuals attract disproportionately more connections than all others. In this figure the blue line shows the correspondence between the share of followers and the share of projects by the top individual. More specifically, the figure shows that in almost one-quarter of projects the top individual gets 100% of all follower links. If we interpret the following as a proxy for interactions, this suggests that in a quarter of projects there are no horizontal interactions between other members. Moving further along this line we see that in over 40% of projects the leading individual gets 50% of all links.[19] The other lines under the blue one show the same relationships for individuals ranked from second to fifth in terms of the follower share received. The figure considers projects involving more than five members. Raising this threshold, the distance between the top individual and the subsequent members becomes larger.

When constructing trade flows, a key decision that we need to make is whether code generated by a person in a given city flows to all other locations from which the project has members, or whether it flows to the city of the owner. Our results presented in Table B1 and Figure B1 provide strong support for the latter approach. Assuming that the code flows to all

[19]We should emphasize that when the leading individual follows others, this also generates a follower link. That implies that even for follower shares below 100% there does not have to be horizontal interaction between project members that are not the leading individual.

Figure B1: The hierarchy of following structures in project teams



Notes: The figure plots the cumulative distribution of the share of followers within projects held by the top 5 team members. The line at the top corresponds to the individual with the highest follow share; the lines below show the follow share of the 2nd, 3rd, 4th and 5th most followed individual.

other cities will vastly exaggerate trade flows because, as suggested by our analysis, many team members do not interact with each other and work independently. To make this more intuitive, we can consider the following example from commodities trade. Imagine that a Chinese phone assembly plant imports separate components from Japan and South Korea. All three countries are thus part of the same supply chain, but the trade volumes generated by this production process do not directly affect bilateral trade between South Korea and Japan, even if all three production units are part of the same multinational company.[20]

[20] In a parallel paper, Goldbeck (2023) is interested in estimating the distance elasticity within the US. The author assumes that every member of a project interacts with every other member in a symmetric way. It is not surprising that, under this assumption, the author obtains a zero distance elasticity. Additionally, the author uses dummy variables at city-pair level, which ignores the intensity of the collaboration both at individual level and how many individuals collaborate between a city-pair. Our approach takes care of the intensive margin.

C Location productivity measures from importer fixed effects

Here we describe a specification in which we recover city-specific productivities from importer fixed effects. In this case we no longer assume that software developers supply labor at a constant marginal disutility. Instead following the conventional model we assume that software developers (workers) supply labor at wage w_i at importing location i. Then the productivity at the city level is given by:

$$T_i = \left(\frac{FE_i}{FE_{SJ}}\right)\left(\frac{w_i}{w_{SJ}}\right)^{\theta} \tag{10}$$

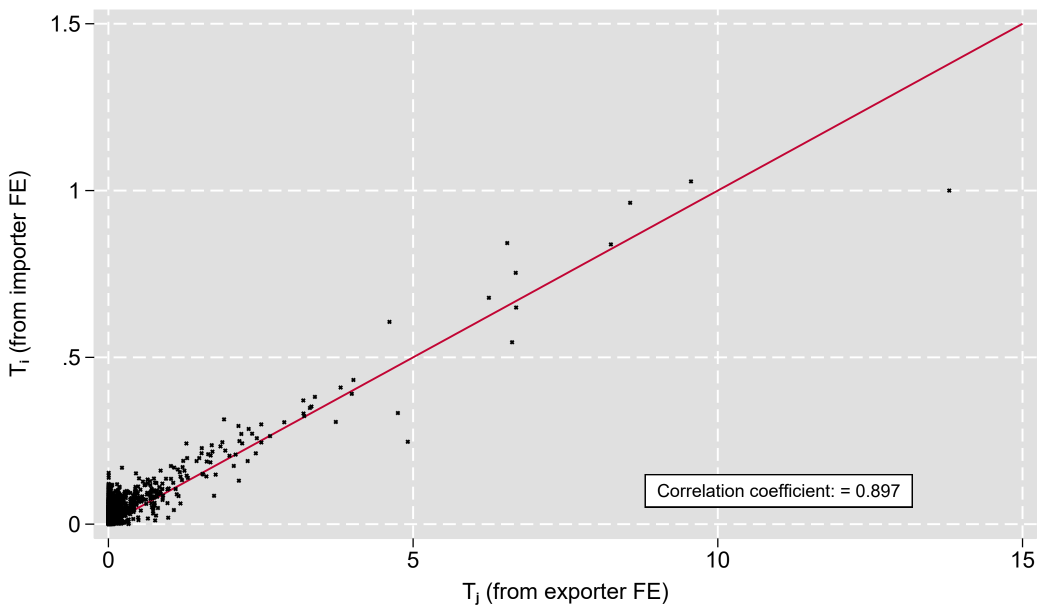
where subscript SJ denotes San Jose, which we use for normalization. Following Waugh (2010) we set $\theta = 0.18$. Because IT specialists' wage data is not globally available at the location level, we construct an approximation utilizing both the ACS and Stack Overflow survey data. To this end, we regress population numbers on average hourly wages for US cities from the ACS data to establish a relation between city size and software developers' average wages. We then estimate country level average hourly wages of software developers by dividing the Stack Overflow country level average yearly compensation of software developers by the average number of hours worked by these IT specialists also from the ACS data, implying that the number of hours worked are uniform across countries. Further assuming that the city-size and wage relationship is constant across countries, we calculate the location level wages as:

$$w_i = \beta_{ACS} * pop_i + w_c \tag{11}$$

where w_c is the country level wage component from the Stack Overflow survey data, β_{ACS} the coefficient from the wage and city size regression and pop_i the population size of city i.

Figure C1 presents a scatter plot of our productivity estimates based on exporter fixed effects against the one based on importer fixed effects with

Figure C1: Correlation of productivity measure from importer and exporter FE



wages. There is a tight fit between both measures with a correlation coefficient of 0.9. Note that the sample is restricted to locations for which both an importer and exporter fixed effect can be derived and to countries for which we have data from the *Stack Overflow* survey.

D Further validation: university rankings

We take advantage of information on the reported affiliations of users. Using this information we construct a ranking of universities. This approach is similar to *Approach 2*. However, instead of aggregating individual scores at the city level, we aggregate individual scores at the university level. More specifically, we identify university affiliated users for the US, the UK and Germany, and sum their individual scores for the identified institutions. Table D1 below lists the top 35 universities that emerge from this approach.

Table D1: Ranking of the top 35 universities in the US, the UK and Germany

Rank	**University**	**Rank**	**University**
1	MIT	19	Cornell University
2	University of California, Berkeley	20	University of Pennsylvania
3	Carnegie Mellon University	21	University of Southern California
4	Stanford University	22	University of Texas
5	Johns Hopkins University	23	University of Edinburgh
6	University of Washington	24	Northeastern University
7	University Cambridge	25	University of California, San Diego
8	New York University	26	University of California, Davis
9	University of London	27	University of Wisconsin–Madison
10	Georgia Tech	28	University of Arizona
11	University of Oxford	29	Princeton University
12	Columbia University	30	Newcastle University
13	Harvard University	31	Imperial College London
14	University of Toronto	32	University of Maryland
15	University of California, Los Angeles	33	Technical University Munich
16	University of Illinois Urbana-Champaign	34	University of British Columbia
17	University of Waterloo	35	University of Utah
18	University of Michigan		

This exercise bears some similarities to the recent paper by Martellini et al. (2024), who use data from the website Glassdoor to construct university rankings. We should emphasize that our ranking is field-specific and includes computer science, mathematics, engineering and some other technical fields whose representatives are intensively involved in computer programming. Also, the ranking does not directly measure the quality of university graduates because individuals with a university affiliation can be faculty members, people working at university labs and students. Even if it only includes faculty members, it is still a valuable measure because it captures the knowledge and contributions of faculty to frontier software projects, which is an important input to the educational process. Importantly, these software projects have real life applications and commercial uses, so our measure does not capture some abstract theoretical knowledge.[21] Compared with the results of Martellini et al. (2024) our ranking is highly correlated with conventional rankings, such as the US News Best Colleges Ranking or the Academic Ranking of World Universities.[22] The fact that the university ranking produced from our data is so closely related to rankings produced by independent sources lends further credibility to our results and indicates that it is unlikely that our data suffers from systematic selection issues.

[21] From this point of view our exercise is also related to Bias and Ma (2023) who construct a distance measure between university course syllabi and academic articles to measure the ”education-innovation gap”.

[22] See http://www.shanghairanking.com/rankings/gras/2021/RS0210 for the 2021 ranking of universities regarding Computer Science and Engineering.

E Additional tables

Table E1: Distance elasticities for trade in tasks by team size

	(1)	(2)
	X_{ij}/X_{ii}	X_{ij}/X_{ii}
Log distance in miles	-0.8993***	-0.6072***
	(0.0856)	(0.0593)
Controls	Yes	Yes
Project team size	<50	≥ 50
Sample	FUA + Admin	FUA + Admin
Observations	11,085,230	222,530
Pseudo R-squared	0.7590	0.7190

Notes: Estimation results of equation 3. Controls include binary dummies for the same country, shared borders and shared official languages. Both specifications are estimated with PPML, and include importer and exporter fixed effects. * (**) (***) indicates significance at the 10 (5) (1) percent level.

F Additional figures

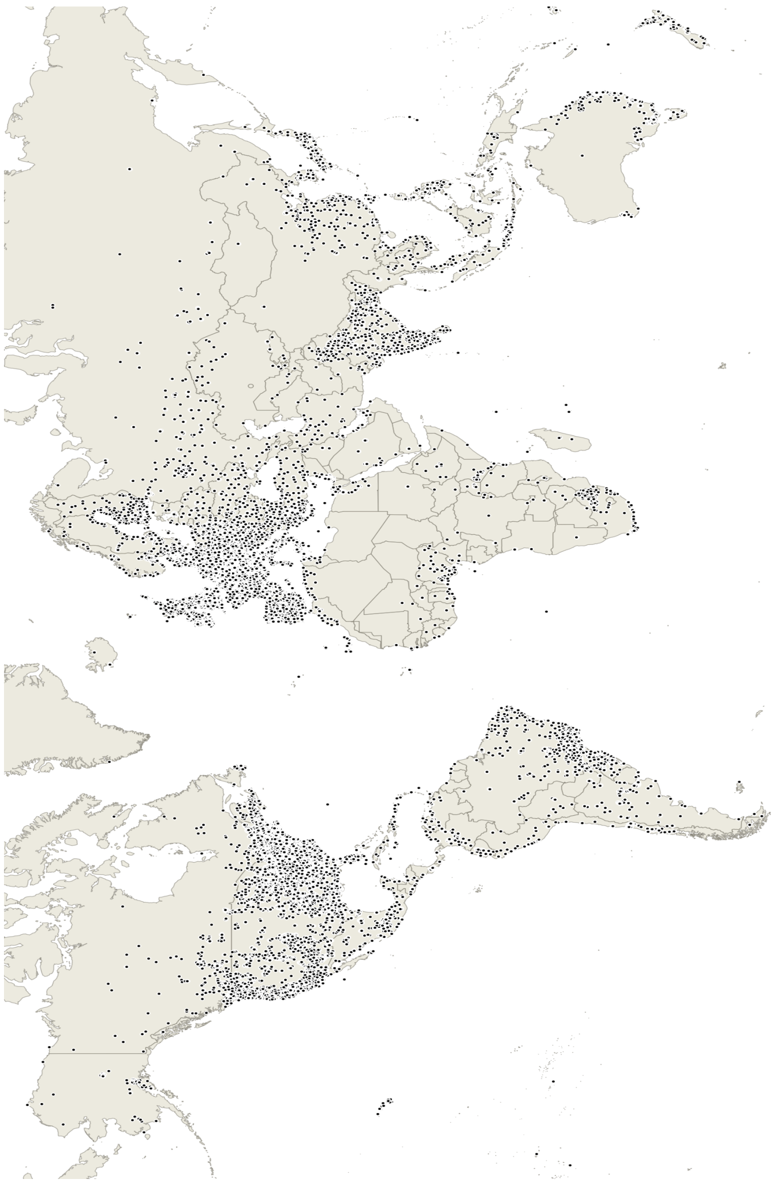


Figure F1: Visualization of GitHub users' locations across the world

Figure F2: Example of sample construction - nightlights (white shading), Functional Urban Areas (blue shading), GitHub users (red dots)

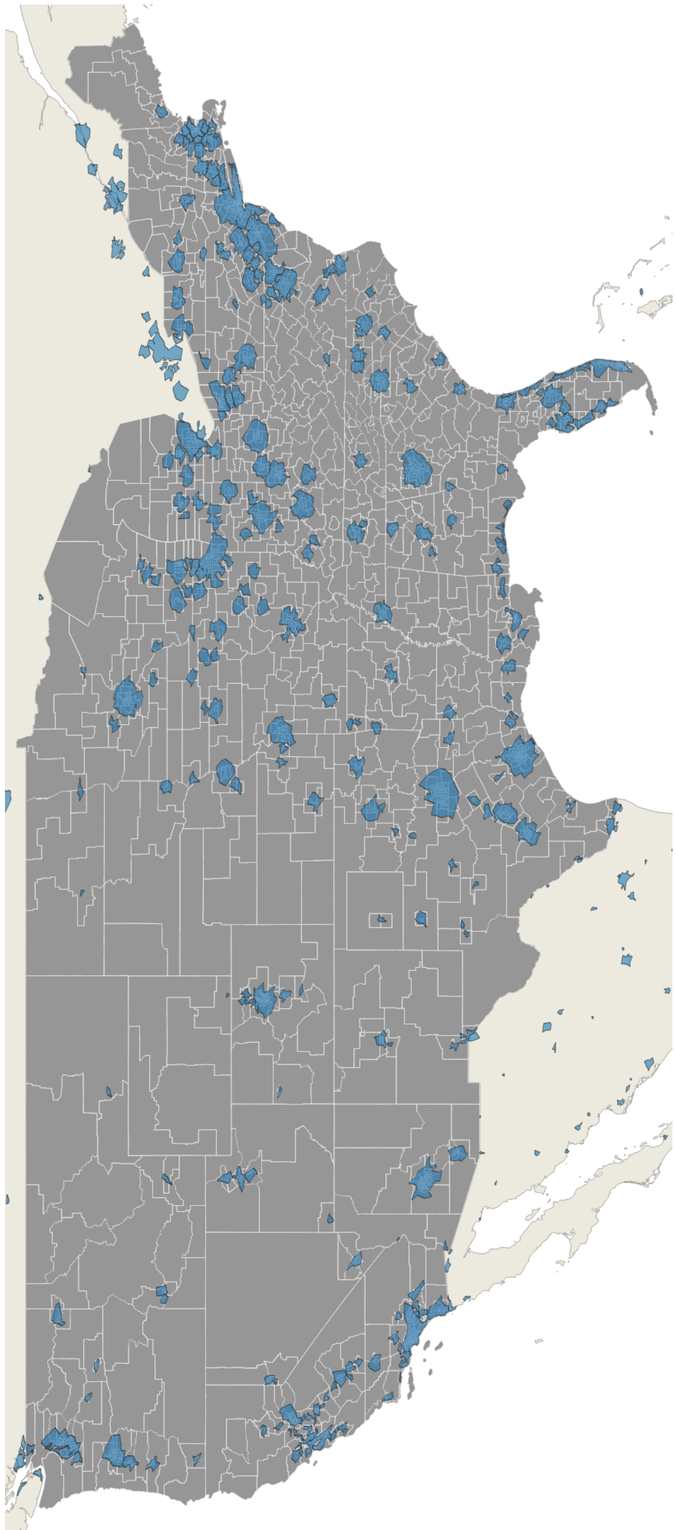


Figure F3: Visualization of the intersection of PUMAs and FUAs.

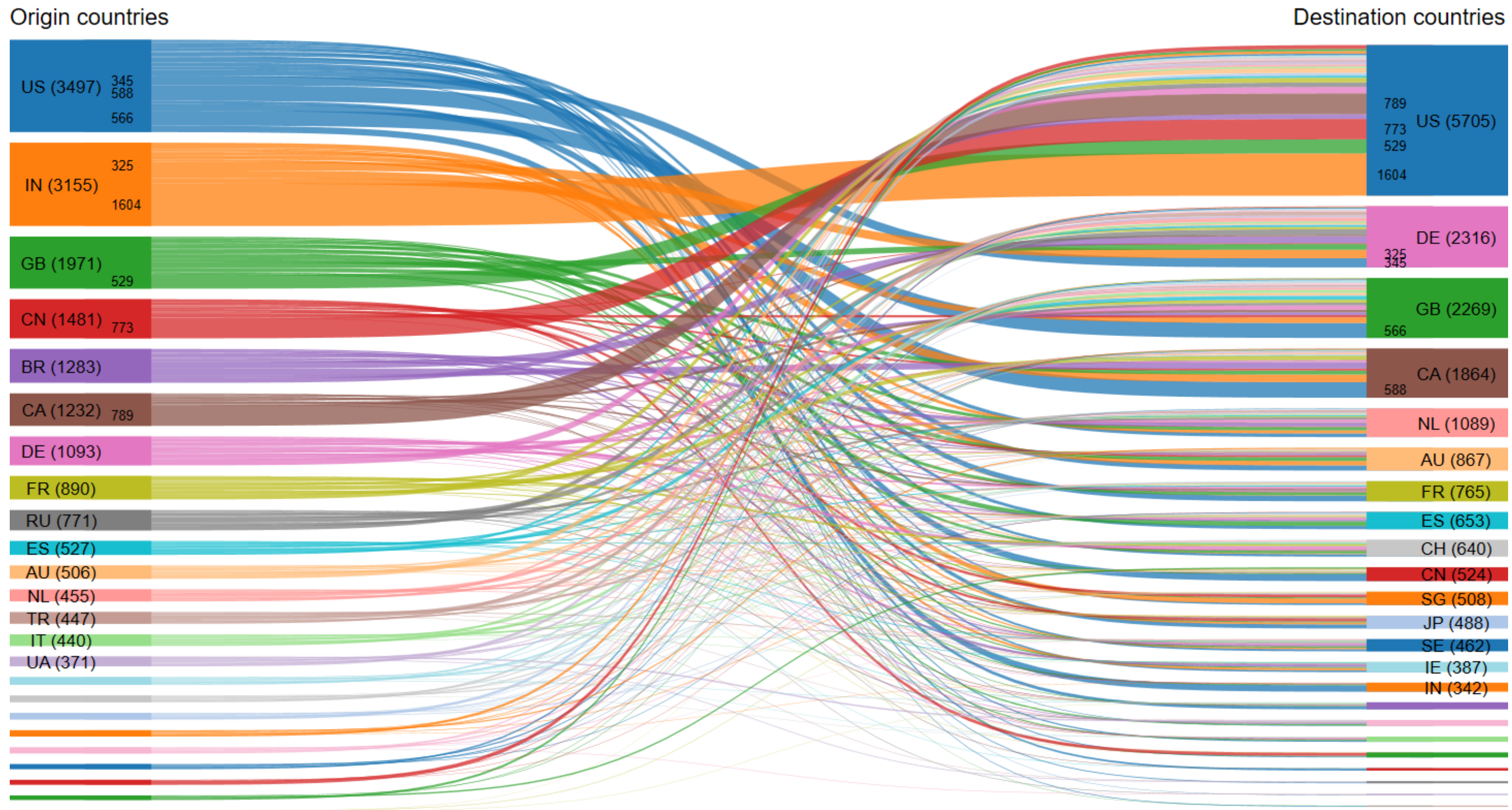


Figure F4: Bilateral migration flows. *Notes: The figure presents the largest bilateral migration flows between origin countries on the left side and destination countries on the right side. For individual flows larger than 300 the numbers in black represent the size of the flow. The numbers in brackets behind the country codes shows the total amount of migrants send or received between the countries displayed.*