The Global Software Production Network

Carlo Birkholz* David Gomtsyan†

December 1, 2024

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 2.55 million software projects across 5,400 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.

*ZEW Mannheim, University of 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 2.55 million projects and 2.64 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 Bengaluru in India and various cities in 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 sec-

tors. 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). Our 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 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.

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.

[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.

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 register 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.

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

[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.

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 (Clemens, 2013; Hendricks and Schoellman, 2017; 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, yet still not fully void of them, or agglomeration effects, as mentioned above.

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

[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).

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 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

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

[5] Commercial open-source software company index.

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

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. This process allows us to eliminate any remaining significant errors in user allocation. We are left with a sample of 2.64 million users with cleaned locations, which is the subset of data we employ whenever our analyses rely on location information. Figure E1 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 so-called 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,

[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".

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' 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 47.3 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 2.55 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

[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.

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:

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

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 rationals 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, which we record accordingly in our data as a flow of commits from the user proposing the alterations to the owner of the parent 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] 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 di-

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

[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.

mensions 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.

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, and income data at sub-national and country level. We describe the construction of all auxiliary data we employ in the Appendix section A.

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 con-

tribution 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 parame-

ter 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

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

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.

4 Results

In this section we start by discussing our estimates of the distance elasticity for the gravity equation and present our estimates of productivity at the city level. We then relate our estimates to nightlights per capita at the city level and GDP per capita at the country level, and compare our estimated productivity gaps between rich and poor countries with macro data. We finish the section by comparing trade in tasks to trade in ideas.

4.1 Structural Estimation Results

In Table 1 we present the results of the gravity equation using PPML. 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 in 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 communication, they cannot fully replace in-person interactions, but rather serve as a complement to them.

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.[11]

In the following columns of Table 1 we report the results for several additional estimations to ensure the robustness of the results. In the second column, we restrict the sample to FUAs and construct the bilateral flows by ignoring users located outside FUAs. The estimated coefficient is not af-

[11] 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 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	-0.8081***	-0.8093***	-0.9129***	-0.6833***	-0.7311***
	(0.0811)	(0.0688)	(0.0834)	(0.1053)	(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	16,678,894	5,266,000	60,945	16,678,894	13,190,040
Pseudo R-squared	0.7067	0.7053	0.8419	0.7087	0.4920

Notes: Estimations 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.

fected. In the third column, we restrict the sample to US FUAs only. The estimated distance elasticity increases slightly, suggesting that there are no large differences between global and US domestic 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. 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 heterogeneity between different commits. 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

[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 -0.858. We conclude that the Covid-19 pandemic does not systematically affect the finding. We also 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 -0.977. This result provides reassurance that the measured distance elasticity reflects indeed spatial frictions, rather than being confounded by temporal misalignments between more distant locations that exacerbate communication challenges.

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 individuals make more valuable commits. Column (5) of Table 1 presents the result for this quality adjusted measure. The resulting absolute value of the distance elasticity is only slightly lower compared to the one in column (1).

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 2. 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, and of Bengaluru, the IT capital of India, lend further credibility to our results. We formally validate our measure in Appendix D by showing significant positive correlations with IT sector wages at the level of US FUAs and globally at the country level. Additionally, 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.

In columns 2 and 3 of Table 2 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 equation 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.

Looking at some individual cities, we can see these differences. For in-

Table 2: Ranking of the top 35 cities across the world

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

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

stance, large cities with many users, such as London or Boston, rank higher in approaches 1 and 2 compared to the rank they receive through the model. Another example is Taichung, which is not a large city compared to other Asian giants but hosts Taiwan's world-beating semiconductor industry. We also find that Poughkeepsie has a relatively high rank. This is the location of IBM's headquarters. The productivity ranking by the model can deliver somewhat unexpected results as well. Specifically, we observe some locations that are not traditionally associated with the IT sector, for example, Las Palmas de Gran Canaria. Such locations might be able to selectively, due to amenities or preferential tax regimes, attract top experts, who can have a profound impact on estimated productivity.

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 3 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

Table 3: 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	0.5248***			
	(0.0634)			
Log GDP per capita		0.8448***	0.8367***	0.9028***
		(0.1162)	(0.1228)	(0.1259)
Sample	FUA	Country level	Country level	Country level
Aggregation method		Average of top 5%	Population weighted	User weighted
Observations	2,639	121	121	121
R-squared	0.0239	0.3252	0.3145	0.3251
F	68.45	52.86	46.45	51.40

Notes: The dependent variables are log productivity estimated from the model. For the country level regressions producitivities 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.

within each country and construct user weighted aggregate productivity at the country level. The results, presented in columns (2)–(4) of Table 3, 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 these 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 4). The equivalent figures are 4.27 log points for within-country population-weighted productivity, 4.15 log points for the average productivity of top 5% locations, and 4.64 log points for GitHub user-weighted

Table 4: 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%	4.15
IT productivity, population weighted	4.27
IT productivity, user weighted	4.64

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 producitivities 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.

productivity. 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 and 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. The productivity gap for industry is 3.71 and for services 3.73, which are smaller than our estimated IT productivity 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 use the second category of forks we described in the data section only, as those capture the dimension of trade in ideas.

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 5 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 2, 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

Table 5: 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***			
	(0.0072)			
Log GDP per capita		0.8082***	0.3396***	0.1280
		(0.1751)	(0.1241)	(0.1048)
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	11,922,149	119	119	119
R-squared	0.6629	0.1363	0.0611	0.0139
F		21.30	7.492	1.493

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.

on GDP per capita. The results of this exercise are presented in columns (2)-(4) of Table 5. 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 turn to the migration of human capital across and within countries. 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.[13] The resulting sample comprises about 1.56 million users, of whom about 98,000 migrate, 38,000 between countries and 60,000 within countries. At the country level, the largest gross outflows of migrants are from the US, India, the UK, Canada and Brazil, while countries with largest gross inflows are the US, the UK, Germany, Canada and the Netherlands. Figure E4 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 6. 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 panels A and B of Table 7, respectively. 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, be-

[13] 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.

Table 6: 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	0.1902***	0.1639***	0.1898***	0.1902***	0.1838***
	(0.0091)	(0.0081)	(0.0052)	(0.0052)	(0.0123)
Observations	939,034	938,552	933,943	921,550	909,621
Pseudo R2	0.0175	0.0630	0.108	0.106	0.222
Panel B:					
2nd quartile	0.6303***	0.5971***	0.6201***	0.6804***	0.5001***
	(0.0224)	(0.0252)	(0.0188)	(0.0160)	(0.0404)
3rd quartile	0.9101***	0.8504***	0.8814***	0.9439***	0.7497***
	(0.0160)	(0.0215)	(0.0218)	(0.0184)	(0.0446)
4th quartile	1.2919***	1.1739***	1.1991***	1.2919***	1.0106***
	(0.0166)	(0.0278)	(0.0279)	(0.0219)	(0.0635)
Observations	1,566,353	1,565,559	1,558,279	1,539,900	1,519,561
Pseudo R2	0.0439	0.0902	0.133	0.123	0.244
Origin country FE	X	X			
Destination country FE		X	X		X
Origin city FE			X	X	X
Number migrants	97,438	97,438	97,438	60,122	37,316

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.

cause we know from the literature that higher-skilled individuals are more mobile (Borjas, Bronars, and Trejo, 1992). In columns (3) and (4) of Table 7, we restrict the sample to migrants only, thereby eliminating potential confounding effects arising from selection into migration. This restriction also addresses concerns that reporting a change in location may be correlated with the quality measure of software developers. In these specifications, the estimated coefficients for upward and downward migration have opposite signs. The results of this table indicate that (i) higher quality software de-

Table 7: Directional migration of individuals based on individual quality

	(1) Up migration	(2) Down migration	(3) Up migration	(4) Down migration
Panel A:				
Log individual score	0.2124***	0.1515***	0.0307***	-0.0343***
	(0.0064)	(0.0081)	(0.0034)	(0.0070)
Observations	872,287	878,591	69,184	66,393
Pseudo R2	0.186	0.128	0.0907	0.127
Panel B:				
2nd quartile	0.6368***	0.5832***	0.0104	-0.0276**
	(0.0214)	(0.0284)	(0.0104)	(0.0119)
3rd quartile	0.9155***	0.8246***	0.0558***	-0.0787***
	(0.0217)	(0.0356)	(0.0091)	(0.0107)
4th quartile	1.2668***	1.0687***	0.0954***	-0.1364***
	(0.0288)	(0.0452)	(0.0101)	(0.0148)
Observations	1,465,610	1,467,499	85,657	82,480
Pseudo R2	0.202	0.147	0.0927	0.131
Destination country FE	X	X	X	X
Origin city FE	X	X	X	X
Sample	All	All	Migrants	Migrants
Number migrants	52,256	37,763	52,256	37,763

Notes: The dependent variable *up migration* (*down migration*) is an indicator variable that is equal to one if an individual migrated to a location more (less) productive than their previous location. In columns (3) and (4) 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.

velopers 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 individuals 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 Ta-

ble 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 columns (3) and (4) 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.

5.1 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 74% of cases in our sample.[14] In columns (2) and (3) we decompose migration into upward and downward migration based on locations' productivities as in Table 7. 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 as in Table 8. The general patterns and estimated coefficients turn out to be very similar to the productivity

[14] We transform the semi-elasticity of 0.3937 according to the following formula: $(100 * (exp(\beta) - 1))$. Multiplying the baseline likelihood of 0.5 with the resulting 48.245% yields around 24% higher likelihood of being above the median quality in the destination.

Table 8: Migration to higher and lower income locations based on individual quality

	(1) Migration to > GDP per capita	(2) Migration to < GDP per capita	(3) Migration to > GDP per capita	(4) Migration to < GDP per capita
Panel A:				
Individual quality	0.3021***	0.1936***	0.0196***	-0.0248***
	(0.0111)	(0.0116)	(0.0040)	(0.0070)
Observations	839,292	807,682	27,416	25,410
Pseudo R2	0.125	0.125	0.141	0.226
Panel B:				
2nd quartile	0.5330***	0.6941***	-0.0086	0.0049
	(0.0306)	(0.0379)	(0.0108)	(0.0153)
3nd quartile	0.8936***	0.9535***	0.0078	-0.0150
	(0.0272)	(0.0368)	(0.0090)	(0.0139)
4nd quartile	1.3681***	1.2778***	0.0344***	-0.0584***
	(0.0268)	(0.0490)	(0.0089)	(0.0150)
Observations	1,393,561	1,345,274	33,800	31,156
Pseudo R2	0.140	0.138	0.142	0.230
Origin city FE	X	X	X	X
Sample	All	All	Cross-country migrants	Cross-country migrants
Number migrants	22,913	14,403	22,913	14,403

Notes: The dependent variable in columns (1) and (3) ((2) and (4)) is an indicator variable that is equal to one if an individual migrated to a country with higher (lower) GDP per capita than their previous location. In columns (3) and (4) we restrict the sample to cross-country 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.

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

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	0.3937***				
	(0.0091)				
Up migration		0.3469***			
(productivity)		(0.0079)			
Down migration			0.4332***		
(productivity)			(0.0134)		
Up migration				0.3284***	
(GDP per capita)				(0.0151)	
Down migration					0.3851***
(GDP per capita)					(0.0155)
Observations	1,560,104	1,553,869	1,553,869	1,560,104	1,560,104
Pseudo R2	0.0050	0.0033	0.0034	0.0025	0.0025
	(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	-0.0496***				
	(0.0125)				
Up migration		-0.1224***			
(productivity)		(0.0121)			
Down migration			0.0561***		
(productivity)			(0.0192)		
Up migration				-0.1449***	
(GDP per capita)				(0.0201)	
Down migration					0.0039
(GDP per capita)					(0.0168)
Observations	1,566,039	1,553,926	1,553,926	1,566,039	1,566,039
R-squared	0.4388	0.1012	0.0714	0.4438	0.4346
Destination city FE	X	X	X	X	X
Number migrants	97,438	52,256	37,763	22,913	14,403

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.

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

5.2 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 loosing 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 appears to be 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. The small positive coefficient for out-migration becomes insignificant for the specification in panel B. We think, however, that the tentatively positive coefficient on out-migration makes intuitively sense, indicating that there is stronger movement in both directions in higher income countries. 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

Table 10: Migration flows at the country level

	(1) Net migration	(2) Out-migration	(3) In-migration
Panel A:			
Log GDP per capita	0.0213*	0.0128**	0.0323**
	(0.0115)	(0.0055)	(0.0129)
Observations	146	146	146
R-squared	0.0177	0.0269	0.0442
Panel B:			
Log GDP per capita	0.0327***	-0.0042	0.0250***
	(0.0075)	(0.0060)	(0.0082)
Observations	108	108	108
R-squared	0.1053	0.0037	0.1028

Notes: In Panel A we require countries to have more than 20 GitHub users. For the outcomes net migration and in-migration 13 countries, and for out-migration 14 countries do not meet this condition. In Panel B we restrict the sample to countries with more than 150 users. Standard errors are robust. * (**) (***) indicates significance at the 10 (5) (1) percent level.

conjecture based on the individual level regressions that wealthier countries are attracting talent, while poorer countries are losing talent.

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 5,400 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 exacerbates existing differences.

These findings present a rather bleak picture for low-income countries. Nevertheless, there are some locations in developing countries, such as Bengaluru, which have very high productivity levels and are ranked among the global leaders. Understanding the evolution of the ICT sector in these places can provide valuable lessons for other locations in developing countries on how to boost productivity in this sector.

There are a number of important questions that require further attention. Follow-up research should, for example, investigate the role of agglomeration effects in the software development sector. Another important question pertains to the potential knowledge spillovers from emigrating software developers back to their origin locations, and whether these spillovers might offset human capital losses from brain drain over the long term. The challenges in tackling these questions involve utilizing a solid identification strategy based on plausibly exogenous shocks, and, in this connection, the need for a longer time horizon. Despite the fact that GitHub has existed as a platform since 2008, the user base was comparatively small in the early periods, such that the utilization of a longer time horizon comes with the trade-off of a much smaller sample size. We believe that it will be possible to answer these questions credibly as more data become available to researchers.

References

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.

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.

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 (2017): "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).[15] 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 remaining paper we use the terms locations and cities interchangeably. We drop locations with less than 10 unique users to avoid calculating very noisy aggregate measures at the location level. The top 20 locations in terms of the number of users are displayed in Table A3. We arrive at a final sample of 5,424 locations in 179 countries. We map all our other data sources into these geographic areas; Figure E2 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 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

[15] 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.

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.

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:

$$Weight_{p,F} = \frac{Share\ intersected\ area_{p,F} * Population_{p}}{Population_{P,F}}, \tag{7}$$

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 E3 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 A2. 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 16409 in the US to 12 in Senegal, Kuwait and Bahrain. The country with the median number of observations has 135 respondents. We winzorise 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 merge this information to our remaining data by 3-letter country codes. From this source we also obtain

data for value added per worker for the industry and services sectors.

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 218,848,238 commits from users whose locations were accurately identified following our data cleaning procedures. Additionally, we identify 380,053,481 commits from users without location information. While this constitutes a share of 36.5%, 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 67.4% 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.

Table A1: Share of local connections by team size

Team size	Observations	Local share
2-5	269,053	0.598
6-20	152,971	0.492
21-100	80,064	0.406
>100	83,041	0.158

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 A2: 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 A3: City user counts

	Location	User count		Location	User count
1	San Jose	101,242	11	Toronto	33,329
2	New York	79,778	12	Guangzhou	32,560
3	London	64,576	13	São Paulo	32,339
4	Bengaluru	62,438	14	Moscos	32,066
5	Beijing	60,909	15	Tokyo	30,909
6	Seattle	46,213	16	Boston	29,773
7	Los Angeles	42,568	17	Chicago	28,983
8	Shanghai	39,951	18	Berlin	23,813
9	Delhi [New Delhi]	38,054	19	Pube	23,221
10	Paris	34,714	20	Seoul	22,137

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:

$$y_{ij} = \alpha + \beta_1 Owner_j + \beta_2 Owner_i + \epsilon_{ij}, \tag{8}$$

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) *i* follows *j*	(2) *i* follows *j*	(3) *i* follows *j*	(4) *i* follows *j*	(5) *i* follows *j*	(6) Share of follows
Owner_j	2.0161***	2.1468***	1.4894***	1.3300***	1.2989***	0.9352***
	(0.0014)	(0.0015)	(0.0028)	(0.0036)	(0.0141)	(0.0018)
Owner_i		1.9697***	1.2169***	1.0627***	-7.2051***	
		(0.0016)	(0.0032)	(0.0041)	(0.9721)	
Same country			0.9506***	0.6787***	0.4621***	
			(0.0018)	(0.0027)	(0.0040)	
Same location				0.4514***	0.2389***	
				(0.0026)	(0.0047)	
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	47,869,198	30,712,310	24,947,588	3,419,080
Pseudo R^2	0.0303	0.0548	0.0517	0.0502	0.0106	0.0323

Notes: Columns (1)-(5) present the estimation results of equation 8, 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

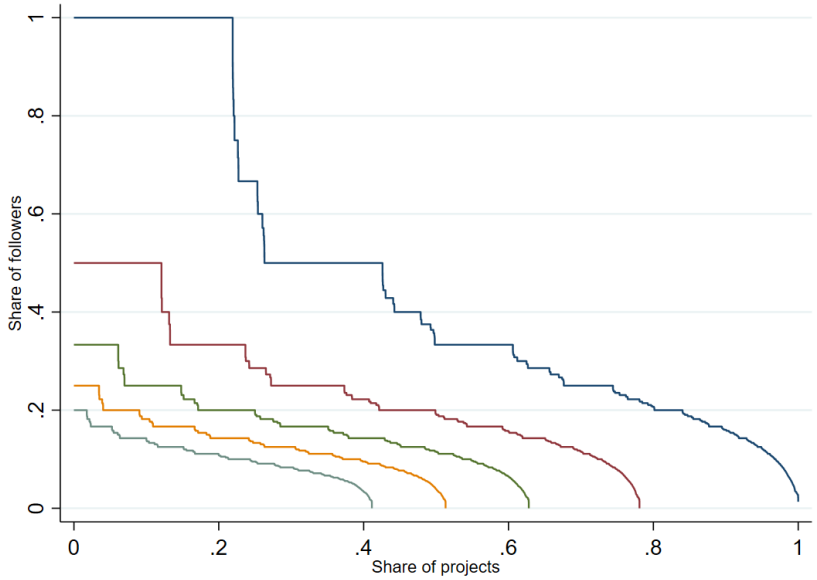
column (4) reveals that in large projects the role of the owner is as central as in smaller projects. In larger projects the owner is much less likely to follow others, which given the larger team size seems to be intuitive. The distinction between large and small projects is important because in our 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 A1). 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.[16] 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

[16] 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.[17]

[17] 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:

$$T_i = \left(\frac{FE_i}{FE_{SJ}}\right)\left(\frac{w_i}{w_{SJ}}\right)^{\theta} \tag{9}$$

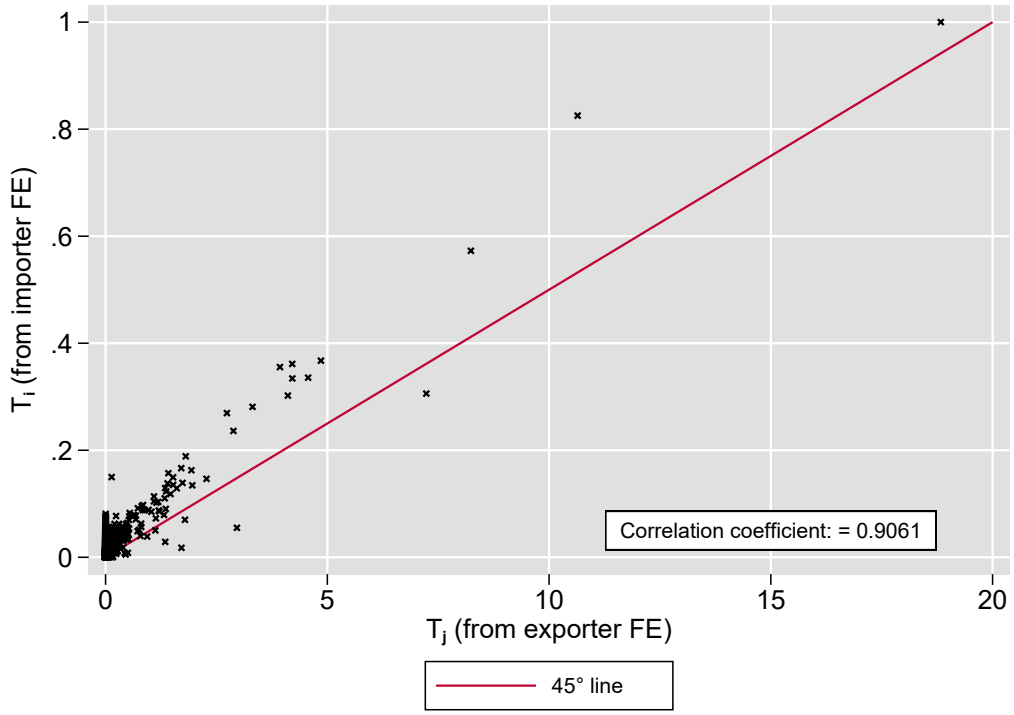
where subscript SJ denotes San Jose, which we use for normalization. Following Waugh (2010) we set $\theta = 0.18$. Because IT specialists' wage data is not globally available at the location level, we construct an approximation utilizing both the ACS and Stack Overflow survey data. To this end, we regress 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:

$$w_i = \beta_{ACS} * pop_i + w_c \tag{10}$$

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 Validation

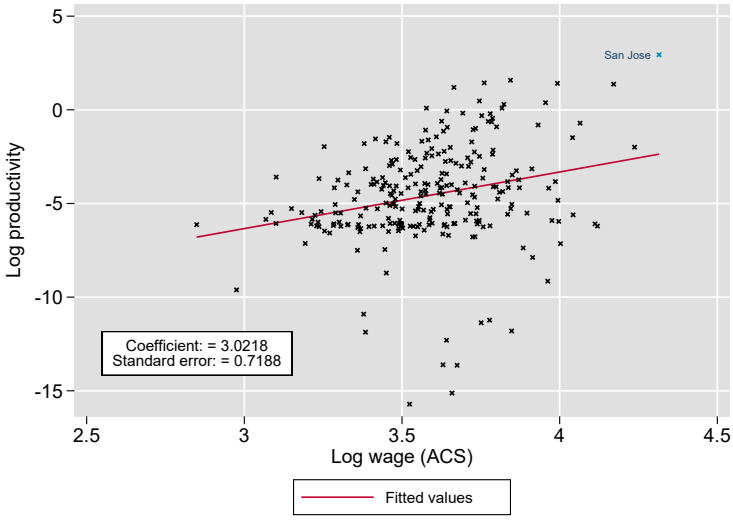
We take two steps to validate our estimated measures. First, we compare our productivity measure with wages. Second, we use our data and construct university rankings and compare them with such rankings from other sources.

Using wages to proxy productivity In the absence of direct measures of productivity, one solution is to use the wages of software developers, which are closely related to productivity, especially in an industry where the share of labor is high.

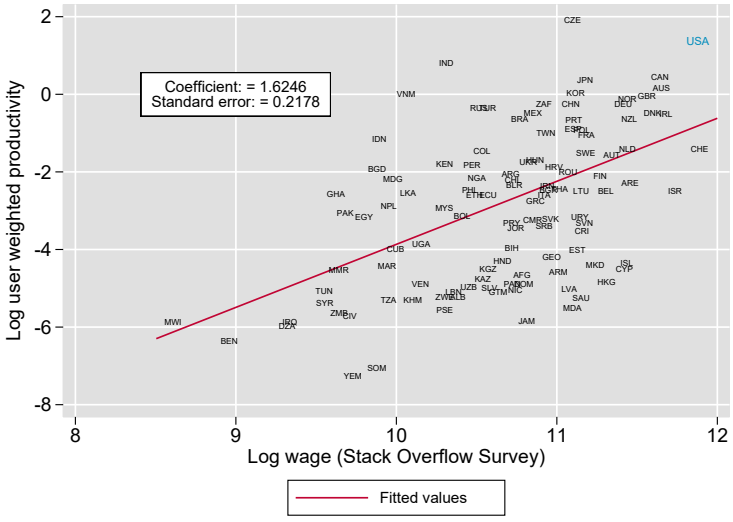
We begin by restricting our sample to the US and regress our productivity measure on the wages of IT specialists in US cities. Our wage data come from the ACS, as described in Section 2. The results are displayed in panel (a) of Figure D1. We observe that both variables move together, also indicated by a significant correlation coefficient of 3.02. In panel (b) of Figure D1 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.

Figure D1: Estimated producitivities 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.

Comparing 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	Northeastern University
2	University of California, Berkeley	20	University of Saarland
3	Carnegie Mellon University	21	Columbia University
4	University of California, Los Angeles	22	University of California, San Diego
5	Stanford University	23	University of Duesseldorf
6	University of Oxford	24	University of Applied Sciences Munich
7	Vanderbilt University	25	Arizona State University
8	Technical University Berlin	26	Harvard University
9	University of Wisconsin-Madison	27	Brown University
10	Johns Hopkins University	28	Purdue University
11	University of Edinburgh	29	California Institute of Technology (Caltech)
12	University of Washington	30	University of California, Davis
13	Cornell University	31	Technical University Munich
14	Brigham Young University	32	University of Cambridge
15	University of Colorado Boulder	33	University of Hawaii
16	University of Arizona	34	University of Essen
17	New York University	35	University of Michigan
18	Washington University in St. Louis		

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.[18] 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.[19] 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.

[18]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".

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

E Additional figures

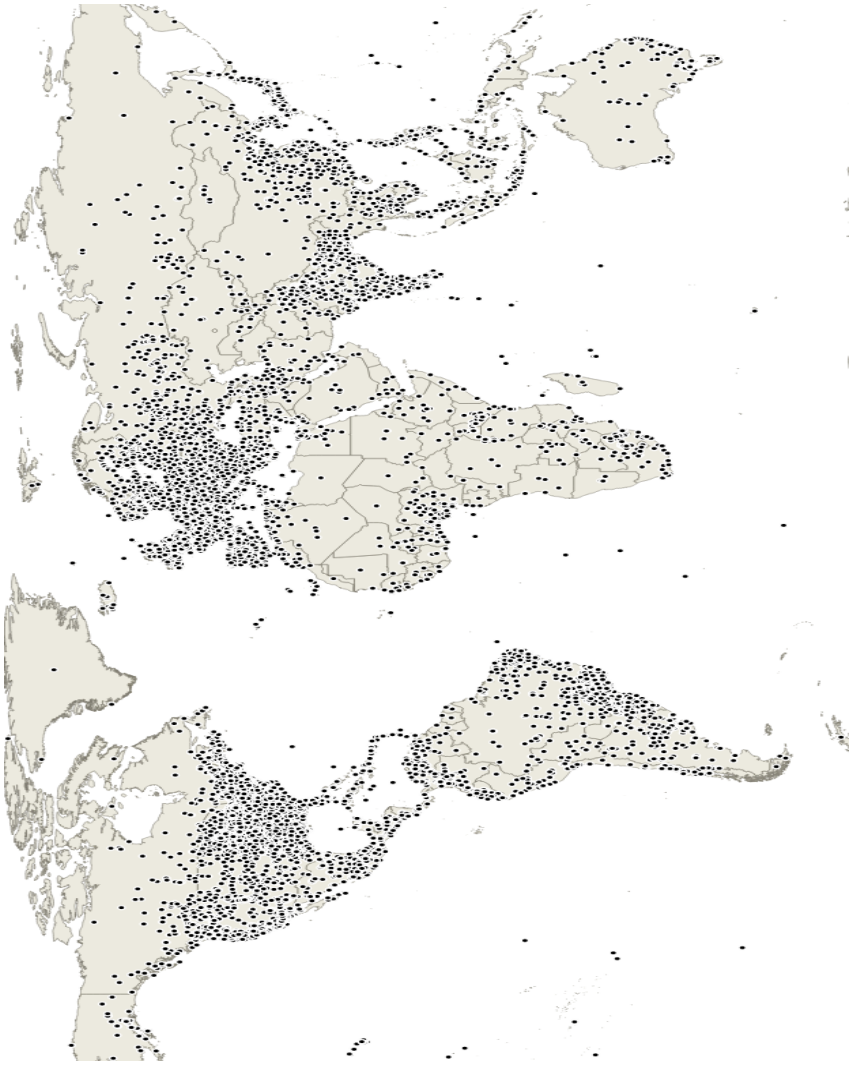


Figure E1: Visualization of GitHub users' locations across the world

Figure E2: Example of sample construction - nightlights (white shading), Functional Urban Areas (blue shading), GitHub users (red dots)

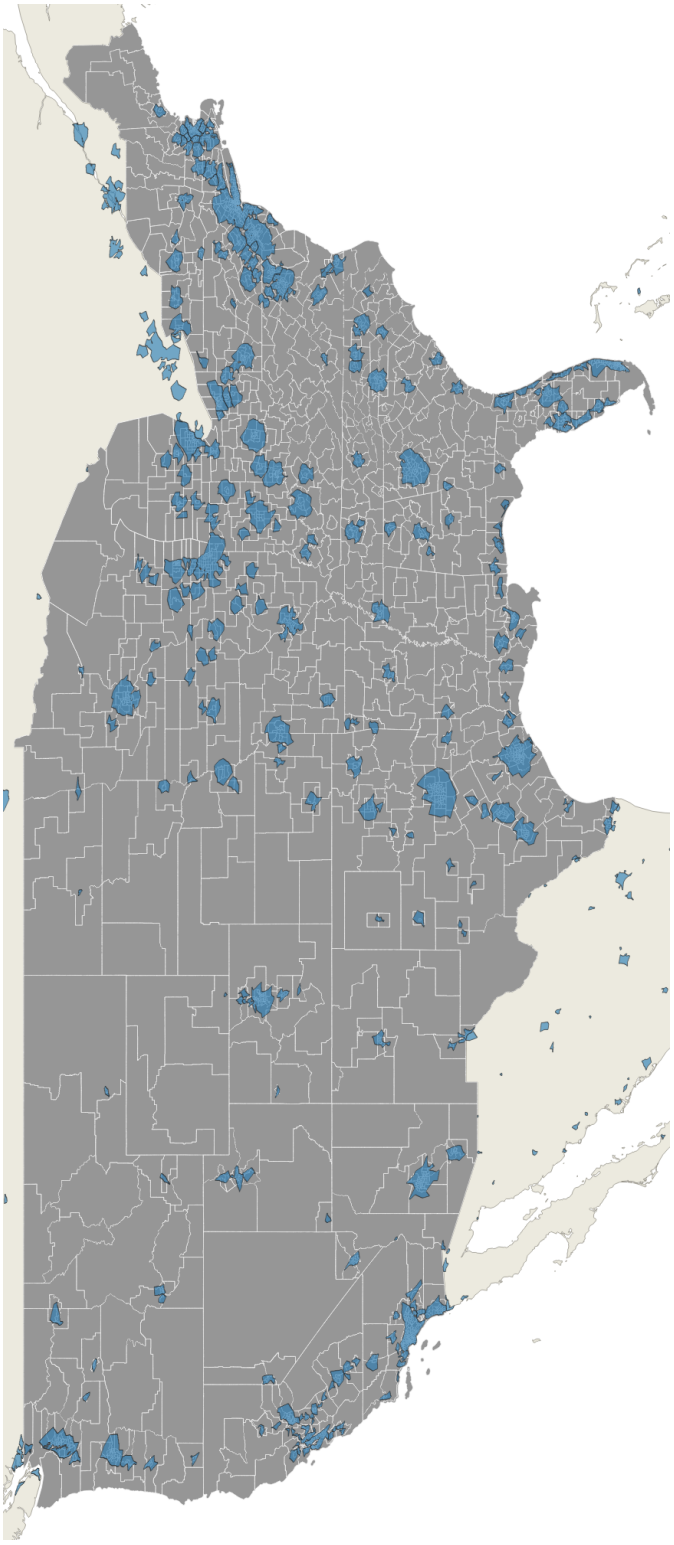
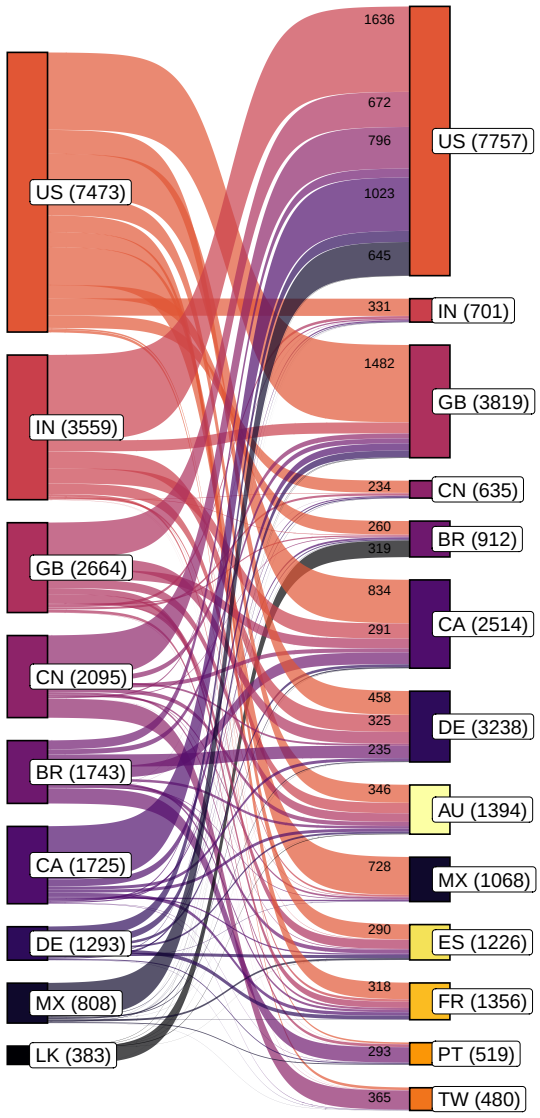


Figure E3: Visualization of the intersection of PUMAs and FUAs.

Figure E4: Bilateral migration flows



Notes: The figure presents bilateral migration flows between origin countries on the left side and destination countries on the right side. We selected all countries that send at least one flow of 200 or more migrants. For the largest individual flows the numbers in black represent the size of the flow. The numbers in brackets behind the country codes signal the total amount of migrants send or received by a country.