geographical analysis

Geographical Analysis (2022) **0**, n/a

Big Code

Sergio J. Rey

Center for Geospatial Sciences and School of Public Policy, University of California, California, USA

Big data, the "new oil" of the modern data science era, has attracted much attention in the GIScience community. However, we have ignored the role of code in enabling the big data revolution in this modern gold rush. Instead, what attention code has received has focused on computational efficiency and scalability issues. In contrast, we have missed the opportunities that the more transformative aspects of code afford as ways to organize our science. These "big code" practices hold the potential for addressing some ill effects of big data that have been rightly criticized, such as algorithmic bias, lack of representation, gatekeeping, and issues of power imbalances in our communities. In this article, I consider areas where lessons from the open source community can help us evolve a more inclusive, generative, and expansive GIScience. These concern best practices for codes of conduct, data pipelines and reproducibility, refactoring our attribution and reward systems, and a reinvention of our pedagogy.

Introduction

Labeled as the "new oil" of the modern data science era, big data has attracted much attention from the Geographic Information Science (GIScience) community. Ignored in this modern gold rush has been the role of *code* in enabling the big data revolution. In this article, I argue that big data is the outcome of transformative processes of *big code* and that we as a discipline need to pay much closer attention to the latter rather than the former.

The article introduces the concept of big code, which expands beyond the specific software package and language of choice to consider the role of community and practices in open source and open science that have revolutionized our world, but from which GIScience seems somewhat disconnected. The article asks how might a deeper engagement with the principles of big code benefit the field of spatial analysis? A central claim of the article is that such an engagement holds the potential for a *better* GIScience.

The article builds on previous themes that I have discussed elsewhere (Rey, 2009, 2014, 2018) examining the intersection of academic GIScience with developments in the broader world

A previous version of this paper was given as the *Geographic Analysis Plenary* at the Association of American Geographers Meeting, April 2021. I thank the participants of that session, and anonymous referees for comments that have improved the paper. Any remaining errors remain my responsibility.

Correspondence: Sergio J. Rey, Center for Geospatial Sciences and School of Public Policy, University of California, California, USA
e-mail: serger@ucr.edu

Submitted: December 22, 2021. Revised version accepted: May 9, 2022.

doi: 10.1111/gean.12330

of open source and open science. It weaves these threads to develop the concept of big code. Rather than viewing code as a means to an end (big data), I see code as a metaphor for how we organize our science and community.

The goals of this article are threefold. First, I introduce the notion of big code and motivate why the concept is of relevance to modern spatial analysis. Second, I surface a number of challenges that our current culture, what I refer to as GIScience Culture 1.0, presents that are holding our field back in terms of the contributions it can make, and the size and diversity of our community. These challenges involve data silos, methods worshiping, and elitist pedagogy. Finally, I identify the specific big code practices that can address these challenges to result in a more expansive, inclusive, reproducible, and enhanced science – in other words, GIScience Culture 2.0.

In the remainder of the article, I first provide a working definition of the phrase "big code" and discuss the motivation for consideration of this concept. Next, I explore how big code intersects with the development of the field of GIScience. Here, I consider some challenges as well as opportunities that the big code era holds for our discipline. The paper concludes with some comments as to what I see as likely future developments.

Big data or big code?

We live in the big data era, and GIScience is no exception. Indeed, concerns with big data have dominated GIScience (Raubal et al., 2018; Li, 2020). Much research has focused on the so-called three V's of big data: volume, velocity, and variety. Addressing the challenges that arise from these characteristics of big data has led to recent advances in cyberGIS, data integration and processing, spatial analytics, and approaches to inference.

While these advances have been impressive, it is hard to miss the sense of a gold rush mentality surrounding much of this research. There are clear echos of an earlier era in the 1990s when GIS was on the ascendancy, and inserting those three letters into a research proposal or job ad seemed like the magic sauce. Today, the term big data plays a similar role. Prominent voices have heralded the excitement around big data:

> Data is the new science. Big Data holds the answers. Gelsinger (2012)

or more dramatically in the claims by Anderson (2008) that the big data deluge will mark the end of theory and indeed the obsolescence of the scientific method itself.

As our field's engagement with big data continues to mature, more reflective perspectives are emerging. I would posit that these correctives take the form of a new set of "V"s to add to big data. Miller (2018) argues that big data is **vanilla** in the sense that its ubiquity can sometimes drive the types of questions that researchers ask, rather than having research questions determine the appropriate type of data. Miller also coined the term *data hubris* to question the myth that ever-increasing data sets will do away with the need for sampling and careful consideration of analytical strategy.

Instead of seeing big data as the "new oil", some argue that it is the new "snake oil":

> you are smarter than your data. Data do not understand causes and effects; humans do. Pearl and Mackenzie (2018)

I would posit two potentially more damaging "V"s to associate with big data: vicious and vacuum. The vicious nature of big data appears in cases where artificial intelligence and

facial recognition programs reinforce racial bias because researchers often train the underlying algorithms on unrepresentative data sets. Who is, and who is not, counted can have draconian implications. The "datafication" process in policing is well known: disproportionate spatial targeting and policing of neighborhoods of color through predictive policing algorithms reinforces and amplifies historical racist policies (O'Neil, 2016).

The big data *vacuum* concerns the issue of who is *doing the counting*. A recent examination of computer science bachelor's degrees awarded by gender in the United States from 1970 to 2010 (D'Ignazio and Klein, 2020, p. 27) has highlighted that the apex for the percentage of degrees received by women occurred in 1984 at 37 percent. Since then, computer science has become ever more the "man factory". Franklin et al. (2021) report a similarly dismal diversity profile for GIScience. The result is *privilege hazard* (D'Ignazio and Klein, 2020, p. 29) whereby the gatekeepers and individuals in positions of power who are designing these AI systems lack the lived experiences of those in the communities bearing the negative consequences of these systems. The resulting systems reflect the biases of the former and are blind to the experiences of the latter.

I am not the first to raise these criticisms of the ills of big data. I do feel, however, that missing from these criticisms is an acknowledgment of the role of a healthy, diverse, and vibrant community in potentially correcting these ills. But, as I discuss more fully below, big code and its associated norms and practices offer ways to foster such communities.

Big code

I suggest that we have largely ignored the role of code in the big data era. Surfacing that role requires a definition of just what *big code* is. I posit three dimensions to the definition of big code:

- Big code as infrastructure,
- Big code as text,
- Big code as conduct/community.

As hard infrastructure, we could rely on traditional software engineering metrics that define "big" in the number of lines of code or person-hours embodied in the code base. These are internal metrics reflecting the input side of the research infrastructure.

We can measure the impact of that hard infrastructure in the consumption of the code as reflected in the download counts for a software project, or in the number of citations the package receives in scientific publications. This assumes, however, that scientific software is given proper attribution (Hwang et al., 2017), and as I return to later, this is a questionable assumption.

These traditional metrics are important gauges, but I would argue they do not encompass the full potential of big code's impact on science. We need a bigger definition of big code. We should expand our view of scientific code as research infrastructure to also consider the role of software projects and communities in the amplification of network effects in the scientific process (Kemper, 2009). Network effects can both grow and deepen the linkages in scientific communities. This view of big code as encompassing not just software but also the community around a scientific ecosystem is similar to the construct of an analytical environment suggested by Bivand (2022).

Code can also improve the quality of the scientific process. Coming to see *code as text* (Rey, 2018) rather than a means to an end can have multiple benefits. Having access to the source code used in the production of a scientific paper increases the transparency and, in turn, the ability to foster replication and reproducibility.

Code as text also has significant pedagogical benefits. Access to the source code can go a long way to demystifying the logic of an analytical method.

A third dimension to the definition of big code is code as conduct. Here, I have in mind the spirit of *Ubuntu* as expressed by Obama (2018):

> "There is a word in South Africa – Ubuntu, a word that captures Mandela's greatest gift: His recognition that we are all bound together in ways that are invisible to the eye; that there is a oneness to humanity; that we achieve ourselves by sharing ourselves with others, and caring for those around us … . He not only embodied Ubuntu, he taught millions to find that truth within themselves."

Code, particularly open source code, is an expression of Ubuntu. Sharing code with the broader scientific community, and doing so in ways that foster a sense of community and collaboration, lead to better science.[1]

Sharing code is a necessary but not sufficient condition for what I view as big code. Many authors of research papers that rely on code for their science often make their scripts available for purposes of replication. However, not all of these come to have communities evolve around the shared code. Big code is thus code that serves as the seed of community growth.

We can view an open source community as an example of a community of inquiry first proposed by the pragmatic philosophers C.S. Peirce and John Dewey. In a community of inquiry, knowledge is viewed as embedded within a social context, and the legitimacy of knowledge stems from inter-subjective agreement among those in the community of inquiry. GIScientists have productively engaged with Pierce's concepts to explore the role of community in improving data semantics (Gahegan and Adams, 2014). In the case of an open source community, it is the (big) code that is embedded within the social context, and the collective contributions of community members enhance the code. But, big code implies more than the function of community in the scientific process. Big code norms and practices are critical for creating these communities in the first place and for their continued thriving and growth.

I see big code as essential to the enabling of modern GIScience. Big data would not have attained its status were it not for the availability of high-quality scientific software and the communities that have evolved around that code. Big code also offers helpful ideas about how we organize our science. It reminds us that we should consider *how* we do our science alongside *what* science we do. Finally, big code means code that is enabling, generative, expansive, and inclusive. In what follows, I expand on each of these themes and how they relate to the current practice of GIScience.

GIScience Culture 1.0

There are aspects of our GIScience culture that, I feel, are holding collective progress back. These pertain to the siloing of data, the worship of method, and an elitist attitude toward pedagogy.

Data silos versus data freeways

In the big data era, the glory is going to researchers consuming big data in their analytical studies. This is somewhat ironic, as such studies would not be possible without access to this data in usable forms. A substantial amount of work happens under the radar that is not given proper attribution in the production of this new scientific knowledge.

A corrective here would see the provisioning of big data as a first-class citizen of the scientific process. We need to move beyond footnote acknowledgment of these contributions to a situation where we no longer view data provisioning as the poor stepchild of analytical work.

Several research groups carry out this vital work of data provisioning. Two prominent examples are the IPUMS group at the University of Minnesota[2] and the Diversity and Disparities group at Brown University.[3] While both groups produce several data products, I focus on the two I am most familiar with in my research. These are the National Historical Geographic Information System,[4] and the Longitudinal Tract Database[5].

Hundreds of studies have relied on these two data sets. Impressive as these contributions have been, it is possible that the impact could be even greater with new models of production along big code and open science practices.

Both projects require login and registration to access the data products. These login/registration procedures allow the project to record statistics on project utilization which, as mentioned above, are essential metrics to demonstrate the project's impact. They also specify the license and citation information surrounding the use of the data.

However, these registration requirements constitute governors on the pace of research flowing from these data sets. The login requirement means that scaling out a research project that uses these data to multiple sites becomes a slow process as each side needs to repeat the registration process. Downloading the data once and then replicating it across sites without first getting permission is a violation of the license agreement for both projects.

Contrast the registration process of these two data projects with the mechanism for interaction with the Linux Kernel project, which is the largest open source project in the world, with 4,189 contributors generating 3,386,347 new lines of code *in 2019 alone*.[6]

The kernel project's impacts have been transformative, reaching into most corners of modern computing from running most of the internet, serving as the core operating system in the majority of the world's mobile phones and tablets, as well as driving the world's fastest supercomputers.

The organization of the project holds important lessons for thinking about how to reinvent GIScience projects. First, through the GitHub repository, the source code is available to *anyone*. There is no registration required to access the code. Second, the repository records the project's entire history, which is also publicly available. Third, by leveraging the strengths of git, the team decentralizes project collaboration.

At first glance, these three features might seem like bugs to someone considering this model for organizing a GIScience software project. However, these features are strengths, not weaknesses. The project's openness enables network effects as new collaborators can contribute with little friction. Recording all activity through the public GitHub repository facilitates reproducibility of the project. It also provides a record of contributions, which internalizes attribution as part of the project's workflow. Rather than complete chaos, the pull request system provides a review process whereby project maintainers evaluate the contributions from community members and determine whether to merge those contributions into the code base. Again, the submission and review are done in the open.

This is a very different development model than in the LTDB and NHGIS projects. By way of example, consider GeoSNAP (Knaap et al., 2019), which is an open source package designed for neighborhood analysis. It implements methods for neighborhood identification, analysis, projections, and visualization. Given the widespread use of the LTDB in neighborhood research, GeoSNAP offers an application programming interface (API) to work with LTDB. However,

because the LTDB license does not allow for redistribution of the data, GeoSNAP provides users with instructions on how to download LTDB to work with GeoSNAP. This additional step is constraining the workflow for users.

Suppose LTDB was organized along the lines of the Linux kernel project with its data as part of a GitHub repository. In that case, GeoSNAP could refactor its API to pull directly from the LTDB repository and remove the manual registration/download step. Moreover, the first time a user calls any functionality in GeoSNAP that requires LTDB data, a request would be automatically made of the download API for LTDB and trigger automatic recording of statistics on utilization. Thus, end-user download statistics would be recorded on the Github repository, providing funding agencies with the vital use metric.

While LTDB does not have a GitHub repository, IPUMS does have a GitHub organization account and related repositories,[7,8] but the projects available appear to focus on packages for working with NHGIS data in different languages (R). Importantly, access to the NHGIS still requires the user to be registered with NHGIS. Users who want to report issues with the underlying datasets or contribute to the infrastructure code that carries out cleaning and integration of the datasets cannot submit pull requests and contribute to this work. The lack of public infrastructure for project development stunts the growth of a developer community around each of the projects.

Expansion of such a community would bring significant benefits to these projects. First, researchers with highly specialized skill sets are maintaining these projects. However, because these teams are small, those same researchers are also responsible for all aspects of project maintenance, and those are significant time commitments. By onboarding new community developers who could take up some of these maintenance contributions, the specialized labor could be more focused on analytical and methodological advances than is currently the case. As a result, the quality and quantity of both scientific code/data and maintenance would increase.

Second, all software projects have bugs. This is a simple reality of complex projects. One way to address this reality is to tap into what Raymond (1999) has termed “Linus’s Law”:

“Given enough eyeballs, all bugs are shallow”.

Growing the developer community around a project is a clear path to having more eyes on the code base and addressing this fundamental issue.

By raising these points, I am not criticizing the work of these two groups (NHGIS, LTDB), who have made valuable contributions to our research infrastructure. Instead, these points are suggestions for expanding the impact of the projects. I recognize that adopting these big code practices is not cost-less and that these small teams are already stretched thin. These transitions, however, are precisely the types of efforts that national funding agencies should be supporting as investments in this infrastructure will pay significant dividends. We could transform data silos into data freeways by adopting big code practices.

Methodolatry and gatekeeping

A second problematic aspect of our GIScience 1.0 culture pertains to what Janesick (1994) has coined as *methodolatry*:

> “a preoccupation with selecting and defending methods to the exclusion of the actual substance of the story being told.”

Our discipline is rife with methodolatry, which takes several flavors. Having served as editor for this journal, the pro-GWR versus anti-GWR split represents an unhealthy schism in

our discipline that resolves around a single method. Similar tribal wars can be seen between the geostatisticians and spatial econometricians, as well as between the Bayesians and the frequentists. Our examples of methodolatry are special cases of a long line of "wars" around the choice of language (R vs. Python), editor (Vim versus Emacs), or even choice of operating system (Windows vs. Mac vs. Linux) seen in the broader computer science world.

Methodolatry results in too much attention being given to the newest shiny methodological innovation and not enough to the substantive applications of these new tools. Methodolatry can also be used to pigeonhole scholars and factionalize the community.

That factionalization can hinder, rather than foster, scientific advances. Instead of tapping into the gains from specialization, the mentality of "not invented here" often raises its head in the form of authoritative gatekeeping. In the early days of open source, the commercial interests of the day were rather dismissive of the new upstart languages. A New York Times article[9] covering the rise of the R language provides an example:

> "I think it addresses a niche market for high-end data analysts that want free, readily available code," said Anne H. Milley, director of technology product marketing at SAS. She adds, "We have customers who build engines for aircraft. I am happy they are not using freeware when I get on a jet."

This gatekeeping also happens within open source communities. For example, in his 2012 SciPy Keynote, John Hunter (Hunter, 2012) recounted the negative pushback he received when first presenting the idea of matplotlib to the Python scientific community. He was told the ecosystem already had visualization packages and did not need another one. Early on, some proponents of the R language claimed that PySAL was guilty of reinventing the wheel. History has demonstrated that these were misplaced criticisms, as matplotlib is now the dominant visualization package in Python. Similarly, the over 1.5 million downloads of PySAL signal the package has filled an essential niche in the Python scientific ecosystem.

Big code opportunities: GIScience 2.0

There are areas where I see opportunities for the GIScience community to innovate by adopting big code practices from the open source world. These pertain to (1) codes of conduct; (2) incentives; and (3) onboarding, pipelines, and education.

Codes of conduct: inclusive participation

A code of conduct is a set of guidelines written by community members that serves to protect the community and inform members of its shared values and norms. The general goal of a code of conduct is to establish and maintain an inclusive culture in the community. Typically, this is through language that offers an open invitation for anyone to participate in a community. Often these guidelines have language that states what is not acceptable behavior. Types of exclusionary behavior that are not tolerated include violent threats or language against another person; sexist, racist, or otherwise discriminatory jokes and language; the posting of sexually explicit or violent material; and the posting of another person's identifying information.[10] Given that software projects encompass both in-person and online interactions, the guidelines extend to both domains.

While the language of a code of conduct is important, mechanisms for reporting and responding to a violation of the guidelines are vital. Without the latter, the former will be

ineffective. A sign of the health of a community is when members feel comfortable enough to report violations and believe that the organization will respond with the appropriate action.

I think there are two underappreciated aspects of codes of conduct. First, there is a recent pushback against codes of conduct by members of the open source community who feel such policies have over-politicized the projects and are stifling innovation (Lynch, 2016). Second, codes of conduct associated with software projects have traditionally focused guidelines governing internal interactions between project members. More recently, these have begun to expand to consider how the project may impact downstream users. The ethics of software development are now appearing more frequently in these guidelines.

Here there are opportunities for engagement between open source practices and academic GIScience. Ethics have long been a focus of discussion in the critical GIS literature (Crampton, 1995), while at the same time, those discussions rarely touched upon interpersonal interactions. The situation has been the opposite in the open source world, so the convergence between the two spheres offers potential synergies for more healthy communities.

In either an open source project or GIScience, we should view a code of conduct as recognizing that participation in open source (science) is often a highly collaborative experience. Therefore, nurturing that collaboration is vital to the success of the enterprise.

Incentives

Incentives play an outsized role in shaping academic communities. By signaling what is valued for career advancement, incentives can guide individuals in deciding how to allocate their energies and time. I see two areas where our incentive system is misaligned with the goal of growing a vibrant GIScience community: dissemination and reproducibility practices, and rewards and attribution norms.

Dissemination and reproducibility

There has been an explosion of interest in these topics driven by the recognition of fundamental crises in many disciplines (Ritchie, 2020). While outright cases of scientific fraud attract most of the attention[11], there are structural features of the culture of science that are arguably the primary sources of the replication crisis. Currently, the published article counts most heavily for career advancement in academia, and publishable means new "significant findings". This results in the so-called "file drawer" effect (Rosenthal, 1979) where a scholar buries null results In a virtual filing cabinet rather than sending the paper to a journal for publication consideration. Thus, there is an inherent publication bias toward "significant findings" in our public record of science. Some have argued that this bias results in most published findings being false (Ioannidis, 2005).

Instead of discouraging the publication of null results, we should be encouraging them. The unspoken rule in academic science seems to be that only studies that reveal discoveries should be published, that is, those results that achieve the magic $p = 0.05$ threshold. Implicit in this view is that findings of null results have zero information and benefit to science. However, there is an actual cost imposed on science from burying null results in file cabinets. Invisible replication of null results may be happening at a scale representing substantial misallocations of research time and cost. If, instead, we publish null results in our main outlets, other researchers may avoid costly rediscovery of unpublished null results and instead focus their energies on new questions and hypotheses.

To encourage the publication of null results, we could adopt promising developments from other fields. Preregistration involves a researcher specifying their research plan in advance of

carrying out their study and submitting it to a registry (Nosek et al., 2018). Preregistration has multiple benefits. First, it separates hypothesis generation from hypothesis testing since the same data are no longer used to both generate and test a hypothesis. The second benefit is that by prespecifying the analytical pathway, preregistration goes a long way toward recognizing the "garden of forking paths" (Gelman and Loken, 2013) problem where each analysis observational studies carry out is contingent on the data employed. Given that there are many possible ways and paths to analyze a particular data set, the one chosen by the researcher must be viewed as only one of all possible paths when evaluating the results obtained.

Improved replication and reproducibility represent the third benefit of preregistration. Because the researcher has specified the analytical path ahead of actually carrying out the data analysis, future researchers can follow the same plan to reproduce the results of the study, or apply the same analytical path to a new data set for replication purposes. In contrast, when preregistration is not adopted, the actual analytical process that yielded the final published results of an article is often unknown to the reader as the authors may have fitted multiple different models, tried various transformations of the data, dropped some observations from the original dataset, or a myriad of possible decisions that are not reported in the published paper. These omissions put reproduction and replication out of reach.

While preregistration can address replication issues, there are two aspects that merit additional attention as we move forward as a discipline. The first is the potential risk posed to junior colleagues who make their work visible at earlier stages only to be scooped by more experienced, and less ethical, community members who can implement and report research results more rapidly. Second, preregistration may not be feasible in all types of empirical research – particularly in exploratory spatial analysis where formal hypotheses are not at hand to preregister.[12] Indeed, hypotheses generation is often an outcome of exploratory work. In these cases, rather than leaving the wider community largely in the dark about what the researcher has actually done, the work could be recorded in `gists` to document the analytical pathways the research has explored. `gists` are a spin-off of Github offering a faster and simpler way to share code online. The lightweight nature of `gists` makes them ideal for exploratory workflows. `gists` can also be made either private or public. One can envisage that in the early phase of exploratory work, the researcher chooses to keep the `gists` private so that they do not run the risk of having their ideas scooped by other researchers. As the work evolves to the point of publication, the relevant `gists` that document the reported research would be made public. This practice would benefit both the individual researcher who can now retrace their steps in what are often multiple and elaborate sequences and future researchers seeking to replicate and build upon the exploratory work.

GIScience is recognizing the problems of reproducibility and replication (Rey, 2014; Goodchild et al., 2021), but it is early days. Our field has adopted more of a stick than a carrot approach toward reproducibility. Journals increasingly require code and data to be made available as a condition for the publication of a research article. This requirement imposes an additional burden on the author as reproducibility is notoriously difficult. The mismatch between true reproducibility, on the one hand, and the necessary condition of providing the source code and data for publication, on the other, results in what I suspect is less than full reproducibility in our field. A possible carrot GIScience journals could adopt is to award badges to manuscripts that are preregistered before submission. In other fields, such badges signal that the authors of the paper are following the community norms of open science.[13]

Badges would be a good step toward increasing the incentives for preregistration and improved reproducibility. However, the burdens of doing full reproducibility and preregistration are still being felt by the individual researcher. A critical need here is community infrastructure to provide resources to lower the costs that individual researchers face when adopting reproducibility practices. GIScience could follow the lead of projects such as the Center for Open Science[14] that provides funding programs to early and mid-career researchers to encourage open science practices among their peers.

Rewards and attribution

While balancing the increasing calls for reproducible research with additional support to carrying out such research is one adjustment we need to take, we can also make changes to our current rewards and attribution practices.

Our discipline has a long history of dismissing researchers who develop tools. When I began my engagement with open source, I was told that tools development was not where I should be spending my time if I wanted to advance in academic GIScience. That was close to 30 years ago. However, this attitude still exists today as reported by Boeing (2020), who recounts an encounter on the job market:

> "You will become known merely as a tool builder rather than a serious scholar. A serious scholar cannot waste time on anything but empirical research and advancing theory."

This kind of attitude is poisonous to innovation in our field. It is troubling to ponder how many methodological advances we may have lost because of these types of encounters driving young scholars away from tools-based research agendas. Rather than penalizing tool builders, our field should be rewarding their contributions (Gahegan, 2019).

One way to reward those contributions is through attribution practices. As mentioned previously, scientific software is not always given proper attribution in our publication processes (Hwang et al., 2017) I suggest two correctives might be effective. First, researchers who use software in their studies should apply the same citation practices for journal articles as for the software. Simply mentioning the package name in the text, a common practice, does not suffice as it fails to register a hook to the citation indices. A second change would be considering software production as evidence of scientific contributions in tenure and promotion cases. The amount of time that goes into a software package that becomes widely used can dwarf the amount of time that goes into a single manuscript. Yet, often, academic developers do not see their code contributions receiving *any* recognition in the evaluation process.

Onboarding: Pipelines to pathways

The lack of diversity in GIScience is a long-standing issue that is now contributing to the aforementioned problems of the big data *vacuum* and the *privilege hazard*. A key aspect of this lack of diversity is the so-called pipeline problem. The claim here is that degrees and tenure track jobs can only go to those who apply, and the lack of diversity in the outcomes reflects a lack of diversity in the applicant pool. In addition to low diversity, our GIScience pipeline also suffers from limited output. Recent industry surveys report difficulty in hiring qualified spatial data scientists (Solem, Kollasch, and Lee, 2013; CARTO, 2020). Our GIScience pipeline is narrow and has a restrictive filter at its opening.

Attempts to address this pipeline problem have been particularly prominent in STEM fields. However, critics have questioned the success of these efforts. Hill (2019) argues that the pipeline metaphor has exacerbated diversity shortcomings. By specifying a set of invariant steps that a student must follow to enter a STEM field, rigid pipelines can exclude students who did not focus on science courses in their high school years.

Alternative models for addressing diversity in GIScience are emerging. Solem et al. (2021) report on an effort to build a research-practice partnership (RPP) (Coburn, Penuel, and Geil, 2013) which brings together university GIScience faculty with high school educators and students, and industry perspectives to create inclusive pathways for careers in GIScience. Focusing on diverse educational communities in Southern California, the RPP collaboratively identified critical problems in computer science and geography at different levels of education, including a lack of awareness about careers in GIScience; challenges in broadening participation within academic and professional geocomputation; and a growing gap between the requirements in the first years of university and the knowledge and skills students possess when graduating from high school.

Through the RPP, the project has five goals: (1) articulate existing curriculum pathways from school to college to career and identify challenges for broadening participation within each pathway; (2) identify the skills and knowledge gaps between geographers, computer scientists, and the geospatial industry; (3) strengthen existing pathways with culturally relevant pedagogy in geocomputation; (4) test those materials for high school and college students; and (5) articulate a replicable framework for establishing RPP's in other states. Principles of big code inform goals 3 and 4 of the project by engaging high school students in the use of open source spatial analysis packages to explore spatial disparities in their communities. This will serve the dual purposes of planting the seed that a career path in GIScience is a possibility, and to introduce and welcome students to the communities around these open source projects.

By transforming pipelines into pathways, programs such as the RPP can influence the future evolution of our discipline. Adopting these big code practices opens up a more inclusive and expansive GIScience community rather than relying on existing reward and promotion systems to reproduce the status quo.

Conclusion

Viewing GIScience through the lens of big code highlights numerous exciting possibilities for reinventing our discipline. An overarching theme in my argument for doing so is the vital role of community and culture in progress. The social practices that have guided the open source movement are catalysts for the success and meteoric rise of big code, the technologies, and their impact.

These are possibilities, but the future is guaranteed to be an open one by no means. Paul Romer, the 2018 recipient of the Nobel Prize in economics for his work on endogenous growth theory and the role of investment in knowledge as a driver of innovation, has recently asked why the open source project Jupyter has succeeded where the proprietary Mathmatica has failed. His answer is:[15]

> "In the larger contest between open and proprietary models, Mathematica versus Jupyter would be a draw if the only concern were their technical accomplishments. In the 1990s, Mathematica opened up an undeniable lead. Now, Jupyter is the unambiguous technical leader.

> The tie-breaker is social, not technical. The more I learn about the open source community, the more I trust its members. The more I learn about proprietary software, the more I worry that objective truth might perish from the earth."

The history of teaching GIScience has an interesting parallel in the tensions between the lock-in and inertia around proprietary GIS packages in university curricula on the one hand, and early attempts at adopting open source approaches and packages in teaching GIScience. Ten years ago, I encountered stiff opposition in moving to QGIS for teaching, and that opposition came from my colleagues as well as university administration. Private discussions with colleagues from other institutions confirm I was not alone in this experience.

The situation has evolved to a point where there is now increasing demand for, and less resistance to,[16] open source curricula. Frequently, proponents point to the "free in beer" nature of open source as the primary driver for this shift, particularly given the budget constraints facing universities.

However, it would be a mistake to focus only on this dimension. Romer's admonition above reminds us that we live in a time when objective truth is under attack, and large social media and software companies have been highly criticized for their complicity in these attacks. Moreover, the closed nature of proprietary software limits the transparency of the underlying algorithms, and the lack of openness is the source of much of the controversy.

The community norms and practices that big code embodies can serve as bulwarks against the ills of closed source software. Those of us teaching GIScience courses within universities who have pushed for open source packages must emphasize these transformative aspects of open source and big code. Instilling those values and practices in our classes and, by consequence, our students will have longer-term positive impacts on society and science that dwarf the short-run benefits to institutional budgets.

Notes

1 See (Ramsey, 2018) for a contrast between open and proprietary stances toward sharing and community.
2 https://www.ipums.org/.
3 https://s4.ad.brown.edu/projects/diversity/index.htm.
4 https://www.nhgis.org/.
5 https://s4.ad.brown.edu/projects/diversity/Researcher/Bridging.htm.
6 https://www.phoronix.com/scan.php?page=news_item&px=Linux-Git-Stats-EOY2019.
7 https://GitHub.com/ipums/nhgisxwalk.
8 https://GitHub.com/mnpopcenter/ipumsr/blob/master/vignettes/ipums-nhgis.Rmd.
9 https://www.nytimes.com/2009/01/07/technology/business-computing/07program.html.
10 For an example of a project code of conduct, see https://GitHub.com/pysal/governance/blob/main/conduct/code_of_conduct.rst.
11 https://research.uh.edu/the-big-idea/university-research-explained/five-cases-of-research-fraud/.
12 I thank a referee for raising these points.
13 https://www.psychologicalscience.org/publications/psychological_science/preregistration.
14 https://www.cos.io.
15 https://paulromer.net/jupyter-mathematica-and-the-future-of-the-research-paper/.
16 In 2021, I was still receiving pushback from colleagues for my use of QGIS and Python, rather than ArcGIS, in my courses.

References

Anderson, C. (2008). "The end of theory: The data deluge makes the scientific method obsolete." *Wired Magazine* 16(7), 1–2.

Bivand, R. (2022). "Analytical Environments." In *Handbook of Spatial Analysis in the Social Sciences*, edited by S. J. Rey and R. Franklin. Cheltenham: Edward Elgar Publishing in press edition.

Boeing, G. (2020). "The Right Tools for the Job: The Case for Spatial Science Tool-Building." *Transactions in GIS* 24(5), 1299–314.

CARTO (2020). The State of Spatial Data Science in Enterprise 2020. Technical report, CARTO, New York.

Coburn, C. E., W. R. Penuel, and K. E. Geil. (2013). Practice Partnerships: A Strategy for Leveraging Research for Educational Improvement in School Districts. William T. Grant Foundation.

Crampton, J. (1995). "The Ethics of GIS." *Cartography and Geographic Information Systems* 22(1), 84–9.

D'Ignazio, C., and L. F. Klein. (2020). *Data Feminism*. Cambridge, MA: MIT Press.

Franklin, R. S., V. Houlden, C. Robinson, D. Arribas-Bel, E. C. Delmelle, U. Demšar, H. J. Miller, and D. O'Sullivan. (2021). "Who Counts? Gender, Gatekeeping, and Quantitative Human Geography." *The Professional Geographer* 73(1), 48–61.

Gahegan, M. (2019). "Our GIS is (Still) Too Small." *15th International Conference on Geocomputation*. Queenstown, New Zealand.

Gahegan, M., and B. Adams. (2014). "Re-Envisioning Data Description Using Peirce's Pragmatics." In *Geographic Information Science* Vol 8728, 142–58, edited by M. Duckham, E. Pebesma, K. Stewart, and A. U. Frank. Cham: Springer International Publishing.

Gelman, A., and E. Loken. (2013). The Garden of Forking Paths: Why Multiple Comparisons Can Be a Problem, Even When There Is No "Fishing Expedition" or "p-Hacking" and the Research Hypothesis Was Posited Ahead of Time. Technical report, Department of Statistics, Columbia University.

Gelsinger, P. (2012). Big Bets on Big Data. https://www.forbes.com/sites/ciocentral/2012/06/22/big-bets-on-big-data/.

Goodchild, M. F., A. S. Fotheringham, P. Kedron, and W. Li. (2021). "Introduction: Forum on Reproducibility and Replicability in Geography." *Annals of the American Association of Geographers* 111(5), 1271–4.

Hill, W. (2019). The Negative Consequences of the Pipeline Metaphor for STEM Fields (Opinion) | Inside Higher Ed. https://www.insidehighered.com/views/2019/10/02/negative-consequences-pipeline-metaphor-stem-fields-opinion.

Hunter, J. (2012). "Matplotlib: Lessons from Middle Age." In Scientific Computing with Python Conference Austin.

Hwang, L., A. Fish, L. Soito, M. Smith, and L. H. Kellogg. (2017). "Software and the Scientist: Coding and Citation Practices in Geodynamics." *Earth and Space Science* 4(11), 670–80.

Ioannidis, J. P. A. (2005). "Why Most Published Research Findings Are False." *PLOS Medicine* 2(8), e124.

Janesick, V. J. (1994). "The Dance of Qualitative Research Design: Metaphor, Methodolatry, and Meaning." In *Handbook of Qualitative Research*, 209–19, edited by N. Denzin and Y. Lincoln. Los Angeles: SAGE Publications.

Kemper, A. (2009). *Valuation of Network Effects in Software Markets: A Complex Networks Approach*. Heidelberg, London: Springer Science & Business Media 2010th edition.

Knaap, E., W. Kang, S. Rey, L. J. Wolf, R. X. Cortes, and S, Han. (2019). *Geosnap: The Geospatial Neighborhood Analysis Package*. https://doi.org/10.5281/zendo.3526163

Li, W. (2020). "GeoAI: Where Machine Learning and Big Data Converge in GIScience." *Journal of Spatial Information Science* 2020(20), 71–7.

Lynch, J. (2016). Are Codes of Conduct Dangerous to Open Source Software Development? https://www.infoworld.com/article/3026196/are-codes-of-conduct-dangerous-to-open-source-software-development.html.

Miller, H. J. (2018). "Spatial Data Analytics." In *Understanding Spatial Media*, 149–57, edited by R. Kitchin, T. P. Lauriault, and M. W. WIlson. London: SAGE Publications, Inc.

Nosek, B. A., C. R. Ebersole, A. C. DeHaven, and D. T. Mellor. (2018). "The Preregistration Revolution." *Proceedings of the National Academy of Sciences* 115(11), 2600–6.

Obama, B. (2018). The Annual Nelson Mandela Lecture.

O'Neil, C. (2016). *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy*. New York: Crown.

Pearl, J., and D. Mackenzie. (2018). *The Book of Why: The New Science of Cause and Effect*, 1st ed. New York: Basic Books.

Ramsey, P. (2018). Esri and Winning. http://blog.cleverelephant.ca/2018/11/esri-dominates.html.

Raubal, M., S. Wang, M. Guo, D. Jonietz, and P. Kiefer. (2018). "Spatial Big Data and Machine Learning in GIScience." In Workshop at GIScience.

Raymond, E. S. (1999). *The Cathedral and the Bazaar*. Sebastopol: O'Reilly.

Rey, S. J. (2009). "Show Me the Code: Spatial Analysis and Open Source." *Journal of Geographical Systems* 11(2), 191–207.

Rey, S. J. (2014). "Open Regional Science." *Annals of Regional Science* 52(3), 825–37.

Rey, S. J. (2018). "Code as Text: Open Source Lessons for Geospatial Research and Education." In *GeoComputational Analysis and Modeling of Regional Systems*, 7–21, edited by J.-C. Thill and S. Dragicevic. Cham: Springer International Publishing.

Ritchie, S. (2020). *Science Fictions: How Fraud, Bias, Negligence, and Hype Undermine the Search for Truth*, first ed. New York: Metropolitan Books; Henry Holt and Company.

Rosenthal, R. (1979). "The File Drawer Problem and Tolerance for Null Results." *Psychological Bulletin* 86(3), 638–6411979.

Solem, M., C. Dony, T. Herman, K. León, A. Magdy, A. Nara, W. Ray, S. Rey, and R. Russell. (2021). "Building Educational Capacity for Inclusive Geocomputation: A Research-Practice Partnership in Southern California." *Journal of Geography* 120(4), 152–9.

Solem, M., A. Kollasch, and J. Lee. (2013). "Career Goals, Pathways and Competencies of Geography Graduate Students in the USA." *Journal of Geography in Higher Education* 37(1), 92–116.