A Need-Finding Study with Users of Geospatial Data

Parker Ziegler
peziegler@cs.berkeley.edu
University of California, Berkeley
Berkeley, California, USA

Sarah E. Chasins
schasins@cs.berkeley.edu
University of California, Berkeley
Berkeley, California, USA

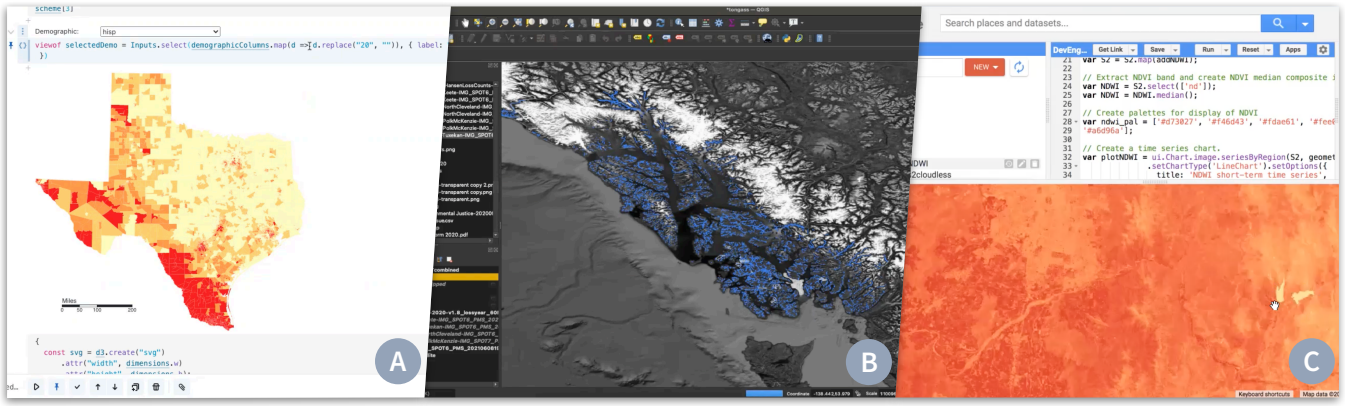


Figure 1: Example screenshots from participants' work with geospatial data. (A) PJ3 creates a choropleth map of Texas' 2021 proposed electoral districts colored by majority racial demographic in Observable. (B) PJ7 combines satellite imagery, stream data, and deforestation data in QGIS to identify illegal logging in southeast Alaska. (C) PE1 computes a Normalized Difference Water Index of their analysis region in Google Earth Engine using multispectral imagery from the Sentinel-2 satellite.

ABSTRACT

Geospatial data is playing an increasingly critical role in the work of Earth and climate scientists, social scientists, and data journalists exploring spatiotemporal change in our environment and societies. However, existing software and programming tools for geospatial analysis and visualization are challenging to learn and difficult to use. The aim of this work is to identify the unmet computing needs of the diverse and expanding community of geospatial data users. We conducted a contextual inquiry study (n = 25) with domain experts using geospatial data in their current work. Through a thematic analysis, we found that participants struggled to (1) find and transform geospatial data to satisfy spatiotemporal constraints, (2) understand the behavior of geospatial operators, (3) track geospatial data provenance, and (4) explore the cartographic design space. These findings suggest design opportunities for developers and designers of geospatial analysis and visualization systems.

CCS CONCEPTS

• **Human-centered computing** → **Human computer interaction (HCI)**; **Empirical studies in HCI**; **Interactive systems and tools**.

This work is licensed under a Creative Commons Attribution International 4.0 License.

CHI '23, April 23–28, 2023, Hamburg, Germany
© 2023 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-9421-5/23/04.
https://doi.org/10.1145/3544548.3581370

KEYWORDS

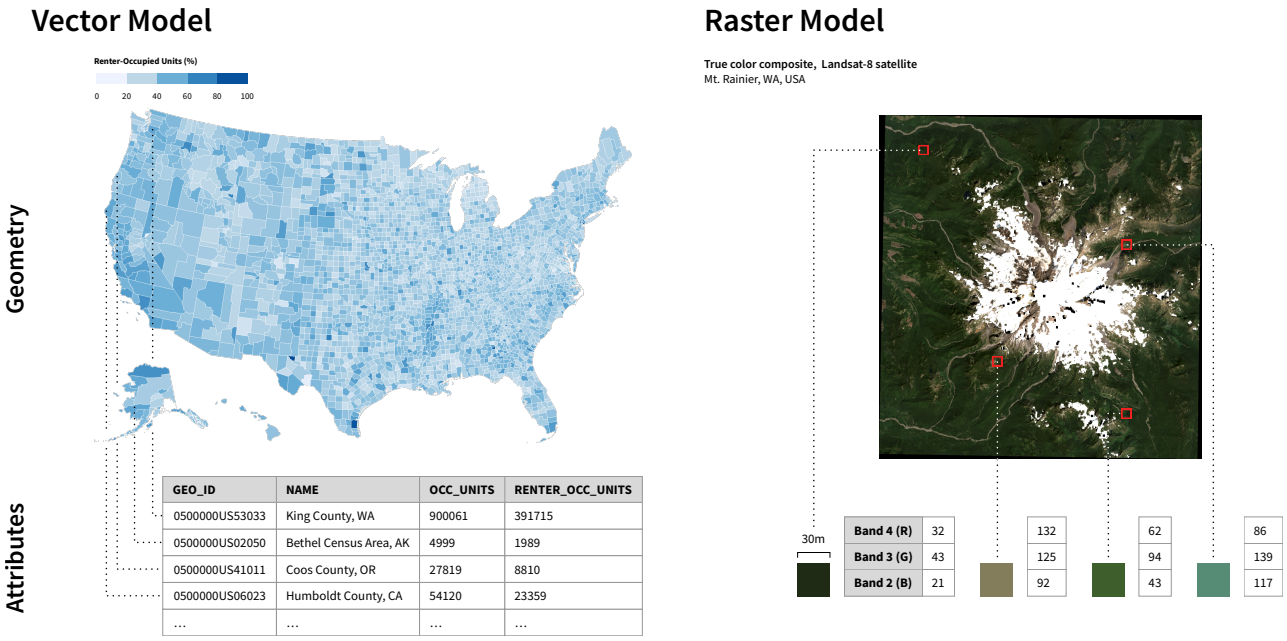
geospatial data, GIS, geography, cartography, contextual inquiry, need-finding

ACM Reference Format:
Parker Ziegler and Sarah E. Chasins. 2023. A Need-Finding Study with Users of Geospatial Data. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23), April 23–28, 2023, Hamburg, Germany*. ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3544548.3581370

1 INTRODUCTION

Geospatial data—data encoding the location and attributes of phenomena on the Earth's surface [59]—is growing in scale and accessibility at a tremendous rate [61]. Researchers estimate that Earth observation satellites generate 80TB of new imagery daily [83]. Closer to the surface, cheap, power-efficient sensors create massive volumes of geolocated data measuring real-time environmental change [40]. Additionally, crowdsourcing efforts like OpenStreetMap have fostered an explosion in publicly available volunteered geographic information [49, 78]. Geospatial data has long played a fundamental role in the research of geographers and cartographers. As this data becomes more available, experts across a widening array of domains are turning to geospatial analysis and visualization to address challenges in climate change [17], public health [34], school segregation [82], hazard modeling [98], and other areas.

Despite this expansion in the community of geospatial data users, research has yet to explore the specific challenges domain experts face in gathering, analyzing, and visualizing geographic information. Many domain experts are self-taught in the theory of



GEO_ID	NAME	OCC_UNITS	RENTER_OCC_UNITS
0500000US53033	King County, WA	900061	391715
0500000US02050	Bethel Census Area, AK	4999	1989
0500000US41011	Coos County, OR	27819	8810
0500000US06023	Humboldt County, CA	54120	23359
...	...	...	...

Figure 2: Geospatial data models. The vector model represents geographies as points, lines, and polygons. Geographies are attached to tabular data via an attribute table. For example, in the choropleth map (left), U.S. counties encoded as polygons are associated with housing data from the U.S. Census Bureau's 2020 American Community Survey. The raster model partitions space into a pixel grid. Each pixel has an attached value corresponding to the data attribute at that location. For example, in the Landsat-8 satellite image of Mt. Rainier (right), each pixel is associated with an RGB value measuring light reflected off the Earth's surface.

geospatial data and the specialized Geographic Information System software used to manipulate it. HCI researchers have found that non-geographers struggle to use these systems because they require familiarity with concepts and terminology from geography [43]. Some of these users have turned to programming as an alternative. While geospatial libraries are increasingly common in Python, R, and JavaScript, domain experts must develop proficiency in at least one of these general-purpose languages to benefit from these abstractions.

Our research aims to investigate the computing needs of the growing community of geospatial data users. Answering calls from HCI researchers for increased collaboration with geography [46, 47], we conducted a contextual inquiry study with 25 geospatial data users from academia, industry, newsrooms, and the public sector. Thematic analysis of observations and semi-structured interviews revealed common challenges across five phases of participants' work with geospatial data: data discovery, data transformation, analysis, analysis representation, and visualization. We observed that participants had difficulty (1) finding and transforming geospatial data to satisfy complex sets of spatiotemporal constraints, (2) understanding the behavior of geospatial operators, (3) tracking geospatial data provenance, and (4) efficiently exploring the cartographic design space, among other challenges. Our findings deepen our understanding of requirements for supporting domain experts in their work with geospatial data and suggest design opportunities for geospatial analysis and visualization systems.

In summary, this paper makes the following contributions:

- A contextual inquiry study of 25 geospatial data users to understand their computing needs
- A thematic analysis of challenges participants faced across distinct phases of their work with geospatial data
- A set of design opportunities for geospatial analysis and visualization systems

2 BACKGROUND

2.1 Geospatial Data

Geospatial data describes the location and attributes of phenomena on the Earth's surface [90]. It differs from tabular data in that it links geometric representations of real-world geographies—referred to as the *geometry* of the data—with attributes of those geographies [59]. In this way, geospatial data connects information to place.

There are two models of geospatial data, distinguished by their geometric representations (Figure 2):

(1) The **vector model** represents geographic features as points, lines, and polygons, connecting tabular data to features via an attribute table. For example, the U.S. Census Bureau's American Community Survey connects demographic estimates to geographic areas (e.g., counties) modeled as polygons [12].

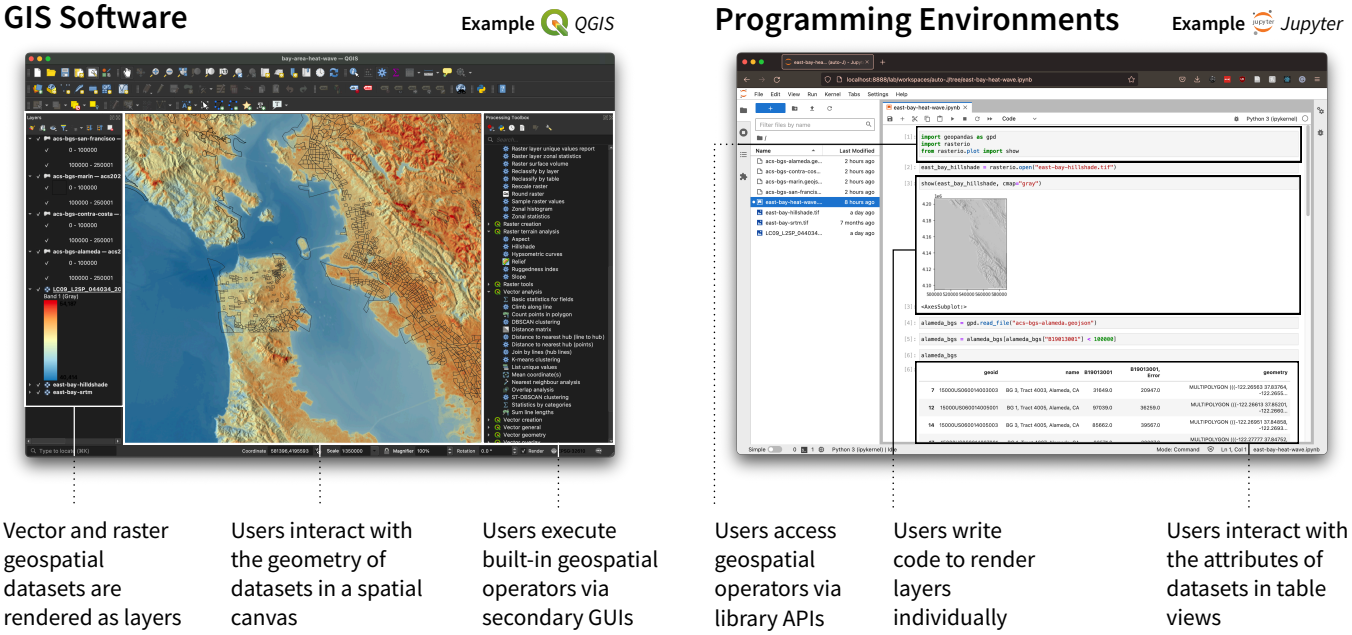


Figure 3: Examples of GIS software and programming environments for working with geospatial data. The QGIS project (left) and Jupyter notebook (right) contain the same geospatial data, but users interact with this data differently in each tool.

(2) The **raster model** partitions geographic space into a pixel grid. Each pixel corresponds to a portion of the Earth's surface depending on the spatial resolution of the raster. For example, the Landsat-8 satellite collects data at 30m spatial resolution, meaning each pixel in the raster represents a 30x30m area [84]. The value associated with a raster pixel corresponds to the data attribute at that location.

2.2 Geographic Information Systems vs. Programming Environments

Geographic Information Systems. A Geographic Information System (GIS) is a software system for "capturing, storing, querying, analyzing, and displaying geospatial data" [14]. GISs represent geospatial datasets as layers, which can be edited, combined, and analyzed to generate new layers using built-in geospatial operators accessed via GUIs. Users visualize and interact with layers in a spatial canvas that allows them to zoom, pan, style, and select geographic features directly. In this way, GISs center interaction with the geometry of geospatial datasets. Interaction with attributes happens in secondary table views where users write SQL to query and manipulate their data. Many GISs exist; our participants used ArcGIS [30] and QGIS [4].

Programming Environments. In contrast to GIS software, programming environments used to work with geospatial data center interaction with the attributes of the data rendered as tables or dataframes. This is especially true of computational notebooks like Jupyter notebooks [79], R Markdown [85], and Observable [74], which have been adopted by geospatial data users but are not purpose-built for geospatial data. In these environments, users write code to visualize and interact with the geometry of their data. Rather than executing geospatial operators via GUIs, they rely on APIs from geospatial analysis and visualization libraries. Newer programming environments like Google Earth Engine [41] and Microsoft Planetary Computer [68] mix features from both GISs and computational notebooks but are designed for particular forms of geospatial analysis (e.g., remote sensing).

3 RELATED WORK

This section surveys findings from observational studies of geospatial data users, empirical evaluations of GIS usability, and studies exploring the needs of data scientists more generally.

3.1 Observational Studies of Geospatial Data Users

Prior observational studies of geospatial data users have focused on identifying GIS usability issues [22, 23, 91, 93]. Our work is most similar to a workplace study of 21 GIS practitioners, which used video recordings, semi-structured interviews, and usability checklists to uncover recurrent participant challenges [23]. The insights centered around error states, finding that GISs failed to prevent common user errors, surfaced errors in difficult-to-understand language, and provided insufficient guidance for correcting errors. Additionally, they observed that non-expert GIS users relied on a "local expert" to perform their analysis, also reported in [28, 37]. Our study differs in two ways. First, we investigate how users interact with geospatial data across tools other than GISs, including computational notebooks, design software, and geospatial analysis and visualization libraries. In fact, most participants (13/25, 52%) did not use GIS software. Second, while [23] observed data transformation

and analysis, we identified additional challenges related to data discovery, analysis representation, and visualization.

Another closely related study observed non-expert GIS users (social science faculty and computer science graduate students) and identified data provenance tracking as a common struggle [93]. Participants' GISs maintained no record of how outputs were generated, making it difficult to reproduce past analyses. Additionally, modifying or retargeting existing maps at new data entailed reverse engineering the original analysis through trial and error. Our study extends our understanding of provenance needs by (1) identifying frustrations with provenance features in modern GISs and (2) describing participants' informal methods for tracking provenance and reproducing past analyses.

3.2 Evaluating GIS Usability

Several studies have evaluated GIS usability using non-observational qualitative methods, including expert task analysis [91], user surveys [21], interviews [29], and screenshot analysis [44]. A task analysis of seven GISs concluded that GIS software is challenging to use because it (1) requires users to understand concepts from multiple disciplines, including geography, cartography, statistics, and databases, and (2) relies on domain-specific vocabulary and concepts (e.g., "overlay," "thematic layer") that reflect the system architecture rather than a user's view of their work [91]. A survey of 159 GIS users found respondents had difficulty understanding and fixing errors, customizing the interface via provided macro languages, and finding sufficient documentation to use GISs [21].

Other studies have employed quantitative methods such as interaction logging [35, 95], controlled experiments [64, 80], and eye-tracking [65, 66] to evaluate particular GIS interfaces. A controlled experiment compared five interaction techniques for cross-layer comparison and correlation [64]. Fechner and colleagues logged interface interactions in a web-based GIS to understand how users collaboratively create and edit geospatial datasets [35]. Unrau and Kray provide a comprehensive survey of studies assessing the usability of different GISs [94]. Rather than evaluating specific GIS interfaces, our study focuses on challenges across various tools.

3.3 Needs of Data Scientists

Research on the needs of data scientists has identified struggles with wrangling and aligning data from multiple sources [26, 71], iterating on and maintaining analysis versions [54, 56], and editing data collaboratively [57]. Data transformation and preparation have consistently emerged as the most challenging phases of data scientists' work [42, 52]; practitioners must develop domain-specific knowledge to identify patterns and anomalies in their datasets, handle missing values, and combine data from differing sources and temporal paradigms [71]. For geospatial data, ensuring that datasets cover the target area and time range of analysis is essential [57]. Beyond data transformation, interviews, surveys, and formative studies have revealed data scientists struggle to track iterations of their analyses, often relying on informal versioning techniques [54, 56]. We examine both challenges—data transformation and version management—in the special case of geospatial data, highlighting areas of overlap and divergence with prior work on data science more broadly.

4 METHOD

To understand the challenges facing geospatial data users, we conducted a contextual inquiry [51] study with 25 participants from academia, industry, newsrooms, and the public sector using geospatial data in their current work.

Participants and Recruitment. We recruited participants via social media (Twitter, Meetup, Reddit, Slack), direct outreach to academic departments, and the authors' networks. We used a screening survey to select participants from multiple domains—including Earth and climate science, the social sciences, and data journalism—with varying years of prior experience working with geospatial data (Figure 4). Our aim with this design was to observe a wide range of user challenges and identify those that recurred across a varied group, revealing needs that transcend domain and expertise boundaries. However, this study design favors breadth at the cost of depth; by prioritizing participant diversity, we may have missed details of challenges that arise only for experts, non-experts, or users in a particular domain. Additionally, recruiting from social media and personal networks runs the risk of creating a more homogeneous participant pool that may not represent the broader community of geospatial data users. Table 1 provides information about our participants.

Consent and Compensation. Before participating in the study, participants signed a consent form in accordance with our institutional review board. Participants received compensation in the form of a $40 gift card or a $40 donation to a 501(c)(3) organization of their choice.

Session Structure. Each study session consisted of a 50-70 minute observation followed by a 15-20 minute semi-structured interview. We conducted sessions remotely over Zoom and recorded them for subsequent analysis. One participant opted out of recording; we analyzed their session via written notes. During observation, we asked participants to share their screen and narrate their thought processes as they worked on a task of their choice related to gathering, analyzing, or visualizing geospatial data. We intentionally left the choice of task open for two reasons:

(1) **Faithfulness to participants' work.** We aimed to study the challenges participants face in their everyday work with geospatial data. Researcher-designed tasks may not elicit the challenges they typically encounter.
(2) **Experience, domain, and tool diversity.** Participants varied widely in their prior experience working with geospatial data, their domain of expertise, and the software and programming environments they use to work with geospatial data. Researcher-designed tasks might lead us to identify erroneous needs that are artifacts of task design.

While the tasks we observed were more representative of participants' actual work than researcher-designed tasks, our study design does not give us insight into how representative they are of the broader community of geospatial data users. Assessing the prevalence of our participants' challenges will require further study.

During semi-structured interviews, we discussed specific observations from the session to confirm or refine our interpretations of participant actions.

ID	Exp. (Years)	Domain	Languages	Tools
PE1	1-3	Earth and Climate Science	JavaScript	Google Earth Engine
PE2	3-5	Earth and Climate Science	R, Python	Google Earth Engine
PE3	<1	Earth and Climate Science	—	QGIS
PE4	1-3	Earth and Climate Science	Python	Google Earth Engine, Jupyter, `geemap`
PE5	<1	Earth and Climate Science	Python	Jupyter, Google My Maps, Leaflet
PE6	5-10	Earth and Climate Science	—	ArcGIS
PE7	>10	Earth and Climate Science	—	QGIS
PE8	3-5	Earth and Climate Science	Matlab	—
PE9	5-10	Earth and Climate Science	Python	Jupyter, `geopandas`
PE10	1-3	Earth and Climate Science	Python	Jupyter, `geopandas`
PS1	1-3	Social Science	—	QGIS, Adobe Illustrator
PS2	>10	Social Science	—	QGIS, Adobe Illustrator
PS3	<1	Social Science	Python	QGIS, Jupyter, `geopandas`
PS4	1-3	Social Science	R	R Markdown, `sf`
PS5	5-10	Social Science	—	ArcGIS
PJ1	1-3	Data Journalism	R	R Markdown, Leaflet
PJ2	<1	Data Journalism	—	QGIS, VisiData
PJ3	5-10	Data Journalism	JavaScript	Observable, D3
PJ4	>10	Data Journalism	Python	Jupyter, `geopandas`
PJ5	5-10	Data Journalism	JavaScript, CSS	QGIS, Adobe Illustrator, Adobe Photoshop, D3
PJ6	1-3	Data Journalism	JavaScript	QGIS, Mapbox
PJ7	3-5	Data Journalism	Python	Jupyter, Microsoft Excel, Tableau
PJ8	1-3	Data Journalism	Python	QGIS, Jupyter, `geopandas`
PO1	3-5	Other (Finance)	—	ArcGIS
PO2	5-10	Other (Computer Science)	Python	IPython, `geopandas`

Table 1: Participant characteristics. Throughout the rest of the paper, we use the participant IDs in the ID column to refer to individual participants. Exp. (Years) refers to participants' prior experience working with geospatial data, in years.

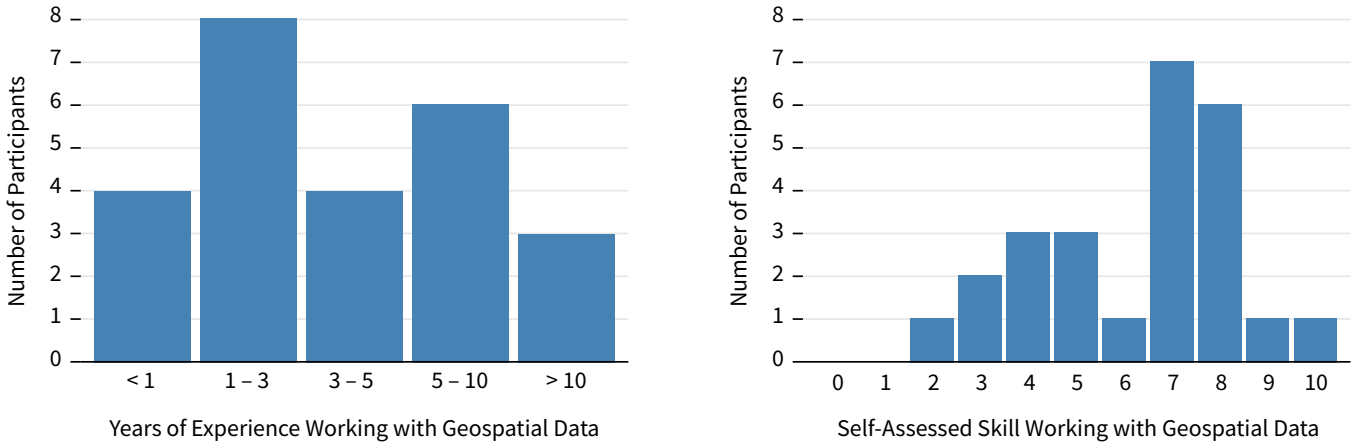


Figure 4: Participant experience and skill. Participants reported their years of prior experience working with geospatial data (left) and their self-assessed skill level working with geospatial data on a scale of 1-10 (right).

Data Analysis. We conducted an inductive thematic analysis [10] of video recordings of the observations and semi-structured interviews using MaxQDA [89]. We started with an open coding phase in which we associated short, descriptive sentences of participant behaviors with segments of the video recordings. We then grouped these open codes into a hierarchy of axial codes and, eventually, top-level themes. The authors met weekly to refine the code hierarchy, splitting and merging open and axial codes based on discussion. We analyzed 29 hours of footage from 24 sessions and written notes from one unrecorded session.

5 FINDINGS

We organize our findings into five sections corresponding to distinct phases of participants' work with geospatial data: data discovery, data transformation, analysis, analysis representation, and visualization.

5.1 Finding Geospatial Data

Participants struggled to find geospatial data satisfying a complex set of spatial and temporal constraints derived from their analysis goals (PE1, PE2, PE3, PE4, PS4, PJ1, PJ2, PJ6, PO2). The most common constraint required that a dataset cover a specific geographic area (PE1, PE3, PE4, PS4, PJ1, PJ6, PJ7). However, it was rare for participants to find existing datasets tailored to their analysis regions. More often, they reduced datasets collected for larger geographic extents by clipping them to their study areas (PE3, PJ7) or filtering out features based on attribute values (PE4, PE8, PE10, PS4). For example, PE3 derived their dataset by clipping a global soil region dataset to their study area and filtering the remaining features by a soil type attribute. In other cases, data for an analysis region was spread across multiple sources and had to be combined manually (PJ1, PJ6). PJ6 traversed 45 pages of the California Air Resource Board's website to obtain the air monitoring boundaries for 15 communities in their analysis region, which they then composed into a single layer. These findings are consistent with prior observations that geographic coverage affects dataset selection [57] and that analysts combine datasets from disparate sources to meet analysis requirements [36, 71].

Some constraints were related to geographic accuracy, which occasionally varied across the analysis region. Accuracy inconsistencies were especially pronounced in crowdsourced geospatial datasets like OpenStreetMap (PE3, PS2, PO2). PO2 explained that in poorly-surveyed areas, "you'll get weird things where building footprints don't fall within block boundaries, or you'll have weird self-intersections … that don't semantically or geographically make sense." These issues were difficult to detect before analysis began due to the size and detail of participants' datasets. Some manually inspected their data to identify and correct topological errors preemptively (PE6, PO1), while others compared their data to satellite imagery (PJ7) or Google Street View images (PJ1) to corroborate its accuracy.

For Earth and climate scientists, constraints on spatial resolution, temporal resolution, and occlusion characteristics of satellite imagery were critical—though difficult—to satisfy (PE1, PE2, PE4, PE8). For example, PE1's analysis of drought patterns in Chile required them to filter Sentinel-2 [27] satellite images of their study area to those captured during the dry season over a six-year period. Occlusions like clouds, mist, and shadows skewed the analysis, prompting them to implement additional image manipulation algorithms to mask the affected pixels.

Participants had additional constraints related to:

- *Cost* (PE2, PE3) – Participants could only use freely-available data.
- *File Format* (PE3, PJ7) – Participants needed data in formats readable by their analysis tool.
- *Programmatic Access* (PE4, PS4) – Participants wanted to query and access data via APIs.

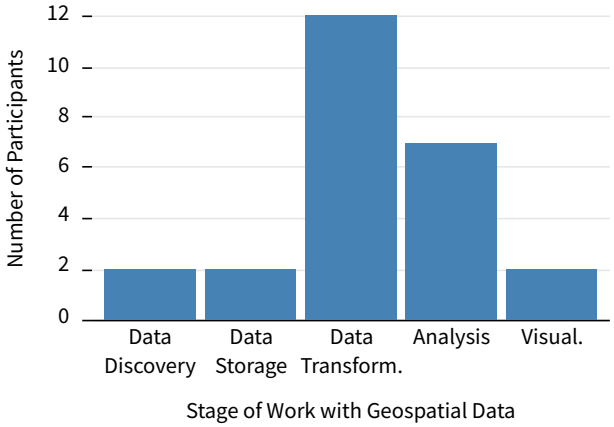


Figure 5: Participant responses to the question, "What part of your work or research with geospatial data feels most difficult?" A plurality of participants (12/25, 48%) selected data transformation, while 28% (7/25) selected analysis.

5.2 Transforming Geospatial Data

Transforming geospatial data was especially challenging for participants, with a plurality (12/25, 48%) reporting this phase most difficult (Figure 5). As PE6 noted, "The data doesn't come all nice, neat, and packaged … The analysis process [can be] pretty thin and bare compared to the preprocessing."

5.2.1 Aligning Geospatial Datasets. Participants needed to align datasets of differing spatial extents, spatial resolutions, temporal resolutions, and areal units to a shared spatial and temporal reference (PE2, PE10, PJ3, PJ6, PJ8). This often required multiple transformations, including resampling, clipping, and spatial and temporal aggregations. For example, PE2's groundwater prediction model used a combination of MOD16 global evapotranspiration data [86] (8-day, 500m), PRISM climate data [20] (monthly, 4km), and USDA-NASS land cover data [96] (yearly, 30m). To align these datasets to a shared spatial and temporal reference, they implemented (1) a resampling algorithm to transform rasters at finer spatial resolutions (30m, 500m) to the coarsest resolution (4km) and (2) an algorithm to accumulate data collected at finer temporal resolutions (8-day, monthly) to the coarsest resolution (yearly). Similarly, PE10 aligned observations from NASA's GRACE satellite to predictions from a land surface model. They created two "masks" in the form of `geopandas` [38] `GeoDataFrames` to filter model predictions to the geographic locations and timestamps for which they had ground truth GRACE observations. In some cases, participants could not align datasets without making approximations.

5.2.2 Topological Errors. Participants spent significant time correcting the topology of their datasets (PE6, PE7, PS2, PO1, PO2). Topological errors refer to violations of geometric invariants such as unclosed polygons, overlapping adjacent polygons, or gaps between adjacent polygons. Most participants identified topological errors through time-consuming visual inspection in GISs or `matplotlib` figures (PE7, PS2, PO1, PO2). PE6 used automated tools in ArcGIS

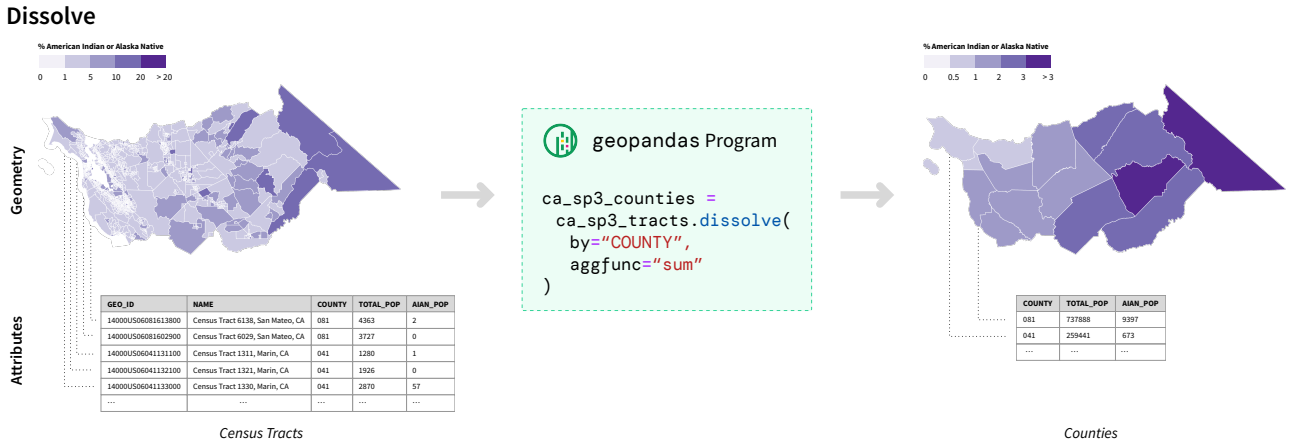


Figure 6: An example geospatial operator. The Dissolve operator is used to group Census tracts in northern California by a shared attribute (`COUNTY`). In addition to merging the geographic boundaries of Census tracts, Dissolve aggregates values in the attribute table using an aggregation function (e.g., `sum`, `first`, etc.).

to find topological errors but explained that fixing these errors required manual intervention.

5.2.3 Reducing Resolution to Improve Performance. Participants faced a trade-off between using data with high geographic precision and spatial resolution and being able to analyze data efficiently (PE3, PJ3, PJ4, PJ5). Greater precision and resolution require more space to encode and, in turn, more compute to process. PE3 balanced this trade-off by using a coarser resolution version of their dataset (≈340km) while iterating on an analysis, even though a higher resolution version (≈1km) was available. This allowed them to experiment with multiple analysis approaches without incurring the performance penalty of processing higher resolution data: "I'm doing it at the lowest definition to just run through the workflow first so I know what I'm doing. I'm probably going to pick a higher definition later." For journalists developing maps for the web, reducing geographic precision minimized the amount of data loaded over users' network connections (PJ3, PJ4, PJ5). PJ3 and PJ4 used MapShaper [9] to simplify the geometry of their vector datasets; in PJ3's case, simplification yielded a 98% decrease in the size of their GeoJSON file.

5.2.4 Data Subsetting and Caching. Participants' datasets were often so large that even analysis and visualization tools purpose-built for this data lagged. "Just waiting for all this to … [render]" (PO2) was a common refrain among participants using both GISs and computational notebooks. PE9, who used `geopandas` to analyze a 3-million point dataset in a Jupyter notebook, waited 50 seconds for a `within` operation to run. PJ2 ran an Overlap Analysis in QGIS between Census block groups and a collection of 2-mile buffers that took six minutes to complete; a previous run using 10-mile buffers "took like two hours." Prior studies have observed GIS users' frustrations with system performance [21, 23], but we found participants using programming environments shared these frustrations.

Some participants accelerated the analysis feedback loop by subsetting data by spatial extent (PO1, PO2). For example, while investigating a bug, PO2 filtered their dataset to features within a subarea of their analysis region. This reduced `matplotlib`'s rendering time from five minutes to one second, allowing them to iterate quickly on a fix. However, they cautioned that this strategy could silence errors when applying the modified code to the full dataset: "We'll subset the entire data universe we're trying to work with and start developing what we think is a generalized tool. And then once we run it on the large universe, we'll find weird inconsistencies and bugs."

Participants also used past outputs as "waypoints" from which they could rerun individual transformations without restarting their entire pipeline (PE7, PJ2, PJ8). For example, PE7 organized the outputs of each preprocessing stage in separate folders ("Level 1 – USGS Product", "Level 2 – Stacked", "Level 3 – MESMA", "Level 4 – Shade Normalized"), explaining:

> I think Levels 3 and 4 [are] where a lot of stuff is going to change, where I might decide to change the parameters or do it a little differently. And so what I can do is just quickly [delete] this entire folder [Level 3] … and it'll clean the slate. And then I'll go back to Level 2, and I'll just rerun everything again from Level 2 to get me to Level 3. … It kind of speeds up the process.

5.3 Analyzing Geospatial Data

For participants, developing geospatial analyses involved constructing pipelines that applied many geospatial operators (in a particular order) to input layers to produce target outputs. Geospatial operators transform both the geometries and attributes of geospatial data, making it difficult to reason about their behavior. For example, the Dissolve operator merges the boundaries of geographic features possessing a shared attribute value and combines attributes of merged features using an aggregation function (e.g., `sum`) (Figure 6). Constructing analysis pipelines required participants to have deep knowledge of operators and their semantics as well as the ability to inspect and debug generated outputs.

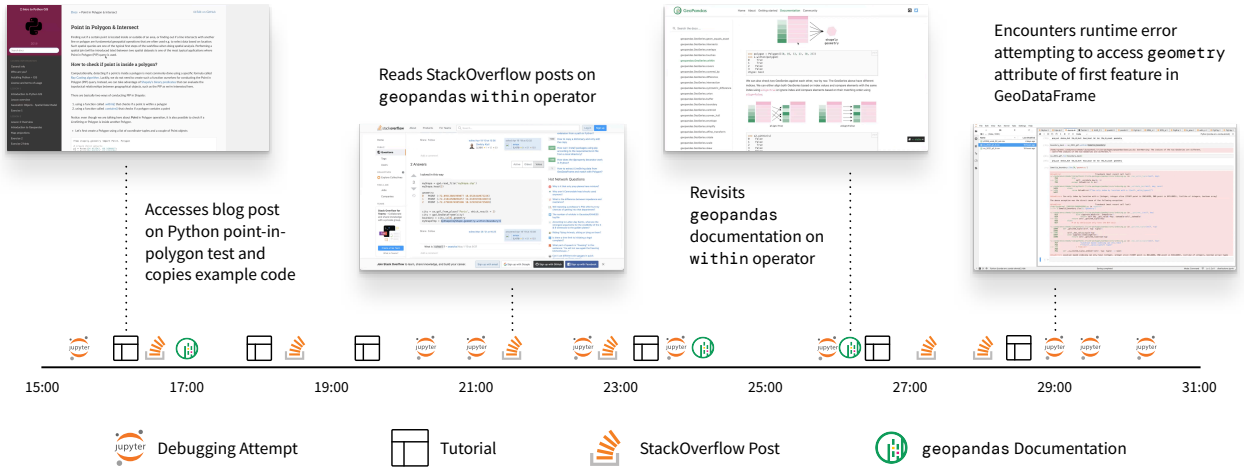


Figure 7: Timeline of PE9's attempts to identify geospatial operators. PE9 moved between tutorials, StackOverflow, and library documentation to identify the correct geopandas operator and syntax to filter a `GeoDataFrame` of cell phone location records to those falling within a polygon from a separate `GeoDataFrame`. They intermixed foraging for example code with testing candidate operators for 16 minutes before arriving at a working solution.

5.3.1 Identifying Geospatial Operators. Participants struggled to identify the correct operators to transform input layers into target outputs (PE3, PE7, PE9, PS4, PJ4, PO2). Even an expert, PE7, noted that distinguishing the behaviors of different geospatial operators is challenging: "I can never remember the vector operations. There's like Union and Merge. Combine! I can never remember exactly what they do. I know exactly what the output should look like in the end; I'm just trying to figure out the tool that gets me that output." PE9 spent 16 minutes searching for a geopandas operator to filter a point layer to locations falling within a specific polygon in a separate layer. They experimented with programs using `intersection` and `sjoin` before identifying a solution using `within`, reflecting: "I feel like I spend a lot of time getting stuck on, like, very simple GIS. It's things like Merge vs. Join, getting confused with which one you want. Or Spatial Join vs. a regular Join. Sometimes just the terminology can be confusing, and sometimes it's not consistent between QGIS and Arc[GIS] and geopandas." The number of operators in GIS software and geospatial analysis libraries exacerbates this challenge. For example, ArcGIS has over 200 operators in its Spatial Analyst toolbox, ranging from bitwise operators to kriging algorithms [31]. This is only one of its 41 toolboxes.

Alternative Expressions of Intent. Although participants struggled to construct analysis pipelines, many could describe their intent in other ways (PE7, PE9, PS4, PJ3, PJ4, PJ7, PJ8). Some used natural language descriptions, either spoken aloud (PE7, PE9, PS4, PJ3, PJ4, PJ7) or written as comments (PE8, PS4, PJ1, PJ4). For example, PJ4 phrased their intent as a question: "How many homicides did each neighborhood have this year, and how did that compare to, like, last year or the last five years, or something like that, right? … So now I'm doing the puzzle in my head, like, how am I gonna get there?" They proceeded to write individual subgoals for each analysis step in comments in their Jupyter notebook (e.g., "Spatially join homicides to [neighbor]hoods"). Some participants interacted directly with features in a map view to express their intent (PS4, PJ7). PJ7 used their mouse to demonstrate in QGIS how they would compute buffers around each line feature in their stream dataset, then compute the area of overlap between these buffers and a raster deforestation dataset. This would yield the total area of illegal logging in their analysis region.

Code Foraging. When participants could not identify the correct operator for an analysis context, they resorted to foraging for similar analysis examples on Google (PE3, PE7, PE9, PJ2, PJ4), StackOverflow (PE9, PE10), in documentation (PE7, PE9, PS4, PJ4), in online tutorials (PE3, PE7, PE9, PS3, PS4, PJ3, PJ8), in colleagues' computational notebooks and source code (PE1, PE4, PE5, PS3, PJ3), or in their own notebooks and source code (PE1, PE9, PE10, PS3, PJ3, PJ4, PJ8). PE9 demonstrated nearly all of these behaviors, visiting six online tutorials, six StackOverflow pages, and two pages of the geopandas documentation to determine the first two operators to use in their pipeline (Figure 7).

5.3.2 Understanding Geospatial Operator Semantics. Even when participants could identify candidate operators, they struggled to understand operator behaviors (PE3, PE7, PE8, PE9, PJ4, PJ8, PO2). As PE7 and PE9 noted in Section 5.3.1, this is partly due to the ambiguous naming of geospatial operators. Moreover, operator semantics differ subtly across GISs and geospatial analysis libraries, meaning "you do need some sort of specificity for doing the actual [analysis]" (PS2) in a particular environment. For example, ArcGIS's Merge combines vector layers of any geometric type—point, line, or polygon—into a single layer [32], while its QGIS-equivalent, Merge Vector Layers, can only merge vector layers of the *same*

geometric type [3]. `geopandas merge` inherits from `pandas`, ignoring geometry altogether and performing a join on shared attributes [39]. As this example illustrates, knowledge of geospatial operator behavior in one tool rarely transfers to another.

Participants used diverse strategies to understand operator semantics. We highlight two common techniques.

Output-Centered Hypothesis Testing. To test hypotheses about candidate operators' behaviors, participants ran operators, then manually inspected generated outputs (PE1, PE3, PE7, PE9, PS1, PJ4, PO2). For example, PE3 attempted to combine two single-band rasters into one multi-band raster in QGIS, hypothesizing that the Merge operator might be appropriate for the task. After running Merge, they inspected the output raster and found that it was still composed of a single band. They next examined pixel values of this raster, noticing they were identical to pixel values of *just one* of the input rasters. From this inspection, they inferred that Merge stitches together input rasters of differing geographic extents rather than combining raster bands.

When testing candidate operators, participants focused on small subsets of pixels or features and compared their values in input layers to their corresponding values in outputs. Sometimes, selection of pixels or features was random (PE7, PS2, PO2). More often, they selected parts of the output where unexpected behavior would produce obviously incorrect values (PE1, PE3, PE8, PS1, PJ2, PJ3). For example, PE1 computed a Normalized Difference Water Index raster and checked the pixel values of a lake in the generated output; if the algorithm succeeded, these values would be close to the maximum value of one.

Observing Feature Count Changes. Many geospatial operations, such as those that filter, intersect, or aggregate geographies, produce output layers containing a different number of features than their inputs. Participants used changes in feature counts to assess operator behavior, with the magnitude and direction of change serving as proxies for correctness (PE9, PE10, PS1, PS4, PJ2, PJ3, PJ4, PJ8). For example, PS4 checked the feature count of the dataframe produced by an `st_join` operation: "This should only be 372 observations because each [Census] tract is unique, but instead `test2` [the output dataframe] is 2790, which is implying that there is something wrong."

5.3.3 Visibility of Geometry in Programming Environments. Participants relied on examining the geometry of their data to understand operator behavior and validate operator output. GISs center the geometry of geospatial data via a map view, a canvas that allows users to pan, zoom, and inspect features and pixels directly. Conversely, participants using programming environments had to write additional code to perform these interactions (PE8, PE10, PJ7, PJ8, PO2). For example, PO2 wrote code to pan and zoom static `matplotlib` figures to particular parcels in their OpenStreetMap dataset. This involved a repetitive process of guessing the coordinates of bounding boxes containing the parcels, updating a Python dictionary encoding these coordinates, re-executing their code in IPython, and re-rendering the `matplotlib` figures until they achieved the desired view.

Programming environments made rendering and interacting with geospatial data challenging enough that, even when participants used them for analysis, they often moved their data to GISs to "see" and "layer" (PJ6) it interactively (PE9, PJ2, PJ4, PJ6). PJ2 explained the immediate visibility of their data's geometry in GISs outweighed the performance benefits of code: "I'm working in QGIS. I know that it's slower than it would be to do it in PostGIS or maybe even `geopandas`, and so I've considered switching to that. But I'm still … new enough that I need to kind of 'see' to make sure my projections are right and stuff like that." PJ4 performed their analysis using `geopandas` in Jupyter but explained they would visualize the results in QGIS: "Now I could try to visualize it here with `matplotlib` and `geopandas`, but I know those things are … not interactive and so I'm like, 'I gotta take this to QGIS.'" These findings extend prior work highlighting visual exploration and cross-layer correlation as integral exploratory analysis techniques for geospatial data users [29, 64]. Participants wanted visibility into their data's geometry not only to identify spatial patterns but also to validate the correctness of their analyses visually.

5.4 Representing Geospatial Analyses

Participants sought to represent their analyses in reproducible, shareable forms. While some GISs maintain a record of users' geoprocessing operations, this record is not grouped by project or user session, omits changes to layer symbologies, and is encoded in formats like XML [33] or command-line expressions [1] that participants did not otherwise use. As a result, participants using GISs created informal program representations outside their GIS to preserve information about analyses.

5.4.1 Reproducing Geospatial Analyses. Participants using GISs had difficulty reproducing their geospatial analyses, either because they struggled to remember the current analysis state (PJ7) or lacked documentation on how they performed the analysis previously (PJ5, PJ6). For example, PJ7 revisited a QGIS project to expand the geographic extent of their analysis region but could not recall if they had already clipped their layers to the new extent. They noted they "come across that problem a lot of remembering where I was and what I've done already." Some participants tried to reverse engineer their workflows from generated artifacts (PJ5, PJ6): "I'm just looking through … some of my previous [exported SVGs] to remember what I did from here" (PJ5). Prior studies of geospatial data users have similarly identified tracking data provenance as a recurrent challenge [93].

Participants using GISs frequently relied on built-in history interfaces to "backtrace" (PS1) operations they ran previously (PE3, PS1, PJ2). For example, PE3 and PS1 used the Recent menu in the QGIS Expression Editor to recover syntax for SQL queries they recently executed, using these as templates to repeat processing steps with modifications. Likewise, PJ2 used the Recently Used menu in the QGIS Processing Toolbox to rerun Buffer and Overlap Analysis operations with new arguments. However, participants explained these history interfaces are limited by how quickly they become overloaded with stale information. PS1: "[I]f you do so many analyses in Q[GIS] in a week, it all gets buried at the end of the day. There's, like, no way you can actually export that history,

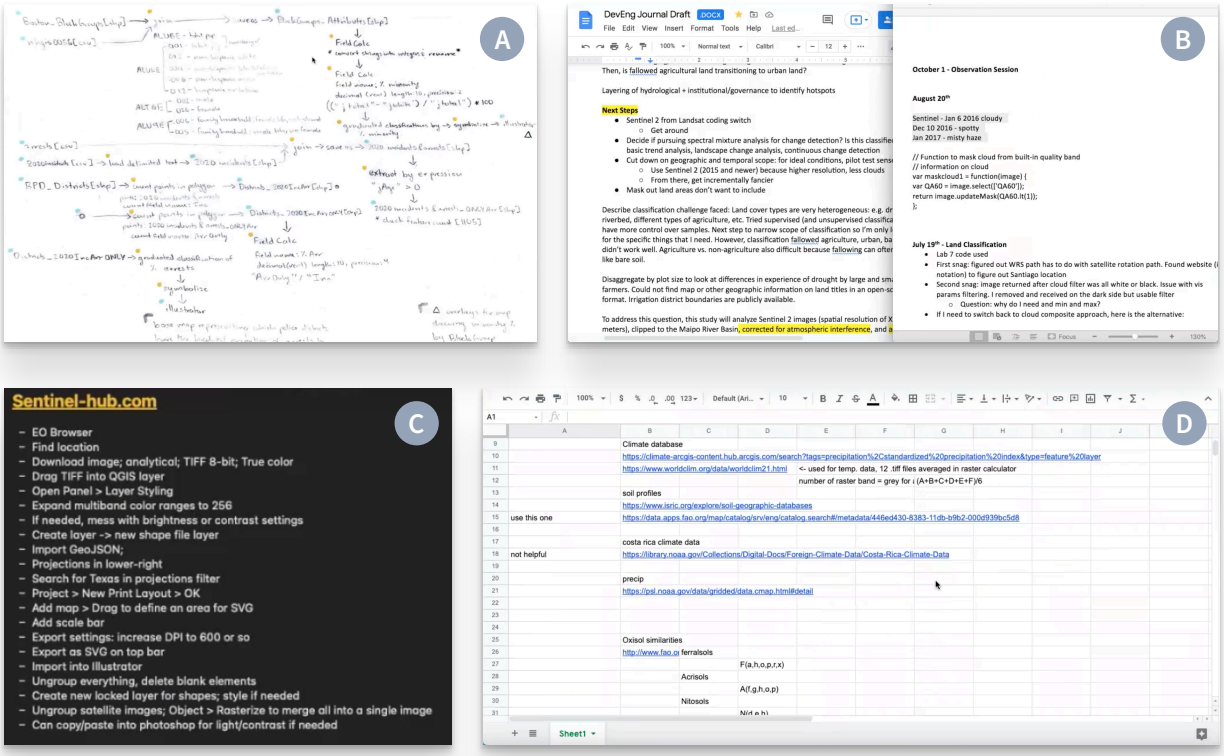


Figure 8: Informal program representations of geospatial analyses. (A) PS1 created a hand-drawn diagram using a custom notational system to record operations, layer names, and debugging steps in their analysis pipeline. (B) PE1 recorded high-level analysis steps, issues encountered during analysis, and code snippets in a Google Doc and Microsoft Word document. (C) PJ5 recorded step-by-step instructions for overlaying a GeoJSON file with a Sentinel-2 image across Sentinel Hub, QGIS, Photoshop, and Illustrator. (D) PE3 used a Google Sheet to record information on data sources, their use in the analysis pipeline, and arguments to pass to operators in QGIS.

which is why I think fundamentally it's only good temporally for a week at most."

Participants using programming environments cited difficulty tracking provenance as a core reason they avoid GISs (PE7, PE9, PS4, PJ8). PE9 explained: "I don't do any of my processing in [QGIS], and mainly because I like that you can track what you did, the traceability of doing it in Python. Versus, there's like none of that if you do it in QGIS. It's like you use a plugin or a function, but there's no track record of it." These participants could also more easily recover information on the current analysis state. While converting a Google Earth Engine pipeline to use imagery from a different satellite, PE1 could not recall how much of this conversion they had completed. To determine where to resume refactoring, they simply ran the program: "I'm just gonna try running this and see what happens because I can't really remember where the part is that I left off." Additionally, participants using programming environments perceived their programs as inherently replicable, shareable artifacts (PE1, PE2, PE7, PS4, PJ1, PJ3, PJ4). PE7: "If someone wants to go back and look at my code—'Oh, he got this shapefile from here and this shapefile from here, and he's pushing them together.' Whereas if you do that in Arc[GIS], you can't really replicate that workflow in the same way."

5.4.2 Creating Informal Program Representations. Participants using GISs created informal program-like representations of their analyses *outside* the software (PE3, PS1, PS2, PJ2, PJ5). Representations ranged from spreadsheets (PE3) to semi-structured text documents (PE1, PS2) (Figure 8). PS1 created a hand-drawn diagram using a custom notation based on the QGIS Graphical Modeler [2]. Their diagram specified: the ordering of geospatial operators; the arguments, input layers, and output layers of each operator; attribute table modifications and validation steps to perform at specific pipeline stages; and layers to symbolize in QGIS before export to Illustrator. They also used color as a visual variable, distinguishing layers from operators using blue and gold dots.

Participants also used informal program representations to record data acquisition, cleaning, analysis, and visualization steps spread across multiple tools (PJ2, PJ5). For example, PJ5 used macOS Notes to document a workflow for overlaying a GeoJSON atop a Sentinel-2 image, which involved moving data across Sentinel Hub, QGIS, Illustrator, and Photoshop. They recorded steps ranging from querying

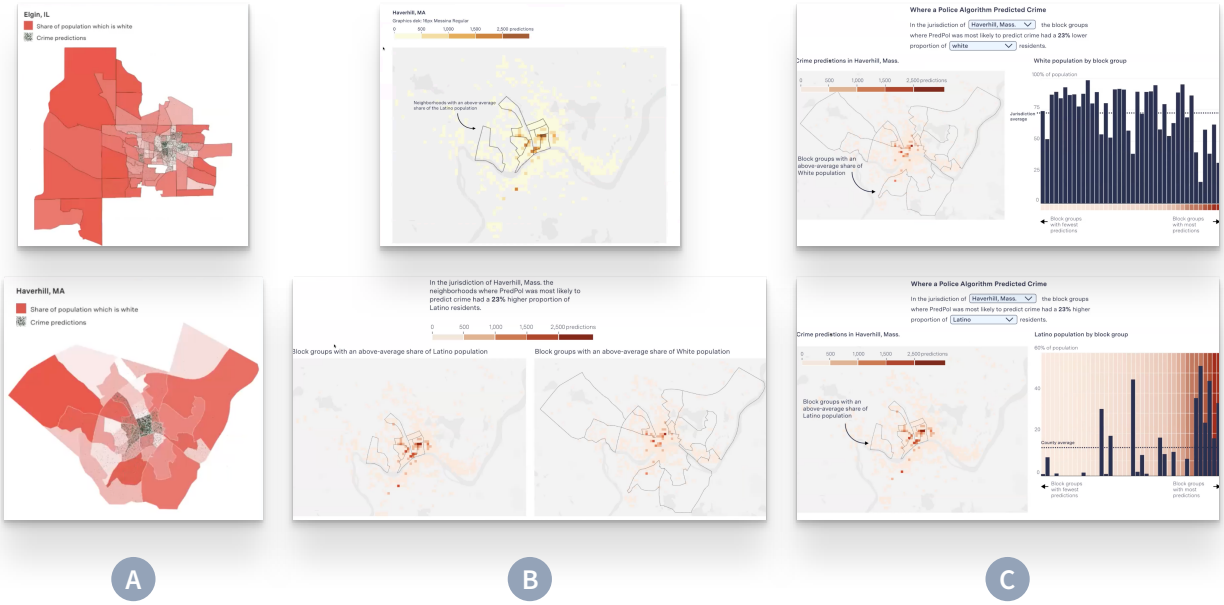


Figure 9: A selection of PJ5's draft maps. (A) PJ5's initial drafts combined choropleth and dot density symbologies in a single map. (B) PJ5 created a gridded heat map (top) and used this style with a modified color scheme in a small multiples layout (bottom). (C) PJ5 tried an alternate layout combining a heat map with a bar chart. These mockups also included a dropdown allowing users to change the variable displayed on the map. The top and bottom mockups show two different UI states in response to user interaction.

a specific Sentinel-2 image in Sentinel Hub to recomposing raster tiles in Illustrator. This degree of tool-hopping was common among participants (PS1, PS2, PJ2, PJ5, PJ6, PJ7, PJ8), but all lacked automated tooling to track cross-system provenance. PJ7 explained: "Y'know, when I'm jumping between QGIS and Python and, well, we were just in Excel, *and* Tableau *and* Adobe Illustrator … y'know, commenting my code in Python doesn't help me remember where I was in Illustrator."

5.5 Visualizing Geospatial Data

Participants wanted to explore an expansive design space of cartographic representations to visualize their analyses' outputs. However, existing tools made this exploration difficult. Visualizing the same data using different cartographic representations involved starting the cartographic design process from scratch for each map variant. Because the tools participants used for map design could not always natively handle geospatial data, spatial information was lost when cartographic work began.

5.5.1 Sketching Cartographic Variants. Participants wanted to visualize data using multiple cartographic representations to explore the design space of possible maps and provide tangible artifacts for collaborators to evaluate (PS2, PJ5, PJ6). PJ5 had over 20 "concept drafts" of maps for one story, ranging from a gridded heat map to a layout combining choropleth and dot density symbologies (Figure 9). One draft included a sequence of mockups showing how the map would respond to a user changing the visualized variables via dropdown menus. These drafts allowed PJ5 to explore cartographic choices with editors: "I have several different versions where someone's like, 'What if this was a fullscreen map and the controls were in the corner, or if this were a side-by-side map?' Yeah, it's predominantly thinking through what is the user experience and what kind of information do we want the reader to be focused on." PJ6 drafted multiple choropleth maps for a story in QGIS and Mapbox Studio, took screenshots of the maps in each tool, designed webpage layouts around the screenshots in Figma, and copied the layouts into a Google Doc for editors to provide comments. They noted that prototyping in a combination of GIS and design software allowed them to compare cartographic choices quickly and get feedback "before I code anything." While these drafts helped PJ5 and PJ6 explore the design space, they were not publication-ready. As a result, both authored code for the chosen maps after the fact.

To create multiple map versions, participants went through their entire visualization pipeline—spread across code, GISs, and design software—for each variant (PJ5, PJ6). Several participants used PJ6's strategy of screenshotting draft maps in GISs (PS2, PE5) or the browser (PJ5) to capture variants at intermediate stages. This allowed them to record many versions distinguished by *minor* cartographic differences (e.g., color scales, basemaps), even if changing map styles was too time-consuming. PJ5 avoided repeating their analysis and visualization process by creating synthetic layers in some mockups: "If you look at those mocks, they're not fully accurate because I wasn't able to do any of the data analysis I wanted to do. So it was more of my crude approximation, like, 'Well, y'know, if we allowed the user to mess with these filters, here's kind of what it would look like.'"

5.5.2 Geospatial Information in Design Software. Participants using GISs for analysis rarely conducted cartographic work there; instead, they moved their data into design software such as Illustrator or Photoshop (PS1, PS2, PJ5, PJ7). This process involved transforming data encoded in geospatial formats (e.g., GeoJSON, GeoTIFF) into non-spatial formats (e.g., SVG, PNG). This transformation jettisons the spatial information of the data, making it challenging to alter analyses after beginning visualization work. Participants described moving from GIS to design software as "crossing a rubicon" (PS2) and "mapping without a net" (PJ5) because the transition broke the link between features and their real-world geographies. For example, PJ5 wanted to alter the brightness of a Sentinel-2 image exported from QGIS to Photoshop while keeping it geographically aligned to a GeoJSON exported separately to Illustrator: "If I crop this by so much as a pixel, right, then it'll no longer be accurate to that geography … But as long as this raster image and my Illustrator SVG remain the same dimensions, then they will be accurate to one another."

Participants used a combination of strategies to avoid spatial information loss when moving to design software. The most common was to avoid resizing a map after export from GIS (PS1, PS2, PJ5), which guaranteed the preservation of spatial accuracy during cartographic design. Participants also exported more data than they planned to use to avoid re-exporting. PS1 maintained a layer group in QGIS named "primary" to house all layers they believed could be important for visualization because "I never know what I want to end up exporting as an SVG into Illustrator."

6 DESIGN OPPORTUNITIES

Our findings suggest new research directions and design opportunities for geospatial analysis and visualization systems.

6.1 Solving Geospatial Data Constraints

Opportunity 1. Participants struggled to find geospatial data satisfying complex spatial and temporal constraints (Section 5.1). While many could describe their constraints succinctly, satisfying them involved constructing bespoke workflows to combine, align, and simplify their raw datasets (Section 5.2). These challenges suggest an opportunity for tools that (1) offer alternative programming abstractions to express data constraints and (2) infer geospatial data queries and transformations from constraints.

Designers could take inspiration from constraint-based programming systems, which have addressed similar problems in visualization [70] and mathematical diagramming [99]. These systems allow users to describe target outputs (e.g., charts, diagrams) via constraints expressed in domain-specific languages (DSLs). Compilers then translate these programs into optimization problems for constraint solvers. In the geospatial setting, GeoSPARQL's topology vocabulary extension [18] provides an example of a constraint-based system for enforcing topological invariants. Our findings suggest that a constraint-based language for geospatial data could allow users to compose additional constraints related to spatial extent, geographic accuracy, spatial resolution, temporal resolution, and occlusion characteristics.

Many constraint-based languages are declarative—users describe *what* a program should generate without specifying *how*. Declarative DSLs have addressed domain experts' needs in fields including cloud infrastructure engineering [45], interactive graphics [48, 88], and programmable biochemistry [77]. A declarative DSL for geospatial data transformation could shift much of the burden of wrangling to automated tooling. For example, we could imagine PE2 replacing their resampling algorithm (Section 5.2.1) with an expression defining the required spatial resolution of all input rasters; the language implementation would be responsible for generating code to perform the resampling. Prior programming languages work targeted at geospatial data [5, 58] has focused on simplifying querying, but our findings indicate that users need better abstractions for transformation.

6.2 Assistive Tools for Constructing Geospatial Analysis Pipelines

Opportunity 2. Participants could describe the target outputs of their geospatial analyses but struggled to construct pipelines to produce them (Section 5.3). This suggests an opportunity for tools that (1) accept non-code specifications of analysis intent, (2) synthesize analysis programs that satisfy specifications, and (3) support users in editing programs.

Program synthesis approaches, such as programming-by-example (PBE) and programming-by-demonstration (PBD), excel in contexts where users can express outputs but struggle to author code. In prior work, C-SPRL used PBD to synthesize spatial data queries from recordings of user interactions in a GIS [92, 93]. However, we are not aware of synthesizers that aid programmers in selecting geospatial operators for their analysis pipelines. Given participants' use of many alternative specifications of intent—natural language descriptions, direct interaction with maps, constraints on outputs (Sections 5.3.1 and 5.3.2)—operator selection may be fertile ground for synthesis. Moreover, prior research in data science has shown that PBE can assist with operator selection in libraries with large API surfaces [7], suggesting that search over the vast numbers of operators in GISs and geospatial analysis libraries is tractable.

Based on participants' code-foraging behaviors (Section 5.3.1), synthesis may be useful even if synthesizers cannot reach all plausible programs. We observed that participants were comfortable tweaking existing programs to reach their target solution. Thus, tools that support goal (3) above may be helpful both as companions to synthesis and independently. This echoes design guidance from [93] arguing that synthesized programs should be editable to support users in adapting them to similar tasks.

Opportunity 3. Participants relied on running operators and manually inspecting outputs to understand operator semantics (Section 5.3.2). This was computationally expensive and time-consuming, suggesting an opportunity for tools that surface information on operator semantics without requiring execution across entire inputs.

Live programming offers users immediate visual feedback on program behavior using concrete inputs [62, 87]. We observed that participants already use small collections of geographic features or pixels as test cases to infer operator behavior, implying that this technique may fit existing debugging patterns (Section 5.3.2). Our

observations of data subsetting practices (Section 5.2.4) reinforce this connection—participants already manually reduce dataset size to get faster feedback. A live programming system for geospatial analysis could automate these practices.

6.3 Reproducible, Shareable Geospatial Workflows

Opportunity 4. Participants using GISs struggled to create reproducible, shareable geospatial workflows (Section 5.4.2). Limitations in existing history interfaces made it difficult to recover information on the current analysis state or revisit past analysis decisions (Section 5.4.1). These struggles suggest opportunities for tools that (1) support efficient search through system history and (2) distill history into a portable and executable representation.

Tools like Verdant [55] and Variolite [54] offer glimpses of alternative ways of surfacing analysis histories. For example, Verdant offers views of computational notebook history by time, artifact, and structured search. It also produces a single-file history representation that users can share with collaborators. Producing usable history tools for geospatial data will first require identifying what history information is valuable. Participants' informal program representations suggest that tracking provenance information at the layer level and recording modifications to layer symbologies and attribute tables could augment existing systems' approach of logging geoprocessing operations (Section 5.4.2).

Beyond making history searchable, developers could borrow techniques from record and replay [13, 63, 67, 81] to make history executable. An observational study of GIS users found that participants' work was often highly repetitive [23], implying that record and replay could help automate tedious tasks in GISs. Although we observed few cases of repetitive work, our participants frequently used history interfaces to manually replay past operations with modified arguments (Section 5.4.1). This suggests an opportunity for tools that can automatically parameterize recordings of user interactions into generalized programs. Ringer [6, 15, 16] and BluePencil [69] are models in this space; for example, Ringer transforms user demonstrations in web browsers into scripts that can be modified, parameterized, and replayed.

6.4 Exploring the Cartographic Design Space

Opportunity 5. Participants wanted to visualize their geospatial data using multiple cartographic representations, but transitioning between representations required engineering each one from scratch (Section 5.5.1). This suggests an opportunity for cartographic design tools that reduce the viscosity [8] of switching between map types.

High-level DSLs offer a low-viscosity approach for design space exploration. In visualization, grammars of graphics [97] like Vega-Lite [88] pair declarative primitives for describing visualizations with a compiler for generating low-level rendering code. Encouragingly, these grammars already have some support for geospatial data. However, because they restrict the geospatial file formats, data models, and cartographic types that users can work with, these grammars cannot express the majority of maps our participants created. In rethinking a grammar of graphics for cartography, our findings indicate that supporting more cartographic representations and minimizing the number of program edits required to switch representations are critical design considerations. Libraries like Plot [75] and Bertin.js [60] are promising examples of tools that make map type a first-class primitive.

Opportunity 6. Many participants used direct manipulation design software to visualize geospatial data. These tools discard all geographic information, making it difficult to refactor an analysis once visualization work has begun (Section 5.5.2). This suggests an opportunity for tools that (1) bridge geospatial analysis and cartographic design and (2) maintain the underlying geospatial data representation of graphical elements while supporting direct manipulation.

Prior work indicates that pairing programmatic and direct manipulation paradigms is possible for tasks with visual outputs. Sketch-n-Sketch [50] successfully applies this technique to SVG editing. Users can manipulate the output SVG or edit the program representation to make changes; Sketch-n-Sketch propagates edits bidirectionally to maintain the correspondence between program and graphic. Such an approach for geospatial data could preserve the geographic information of graphical elements during visualization, allowing users to return to analysis without obviating in-progress design work. Moreover, this approach could address participants' core issue with using direct manipulation design software for cartography—that once a particular map design was chosen, they often had to reproduce the map in code (Section 5.5.1).

7 LIMITATIONS AND FUTURE WORK

In this study, we identified shared challenges and computing needs of geospatial data users across disciplinary and expertise boundaries. We expect future research will uncover additional needs beyond our selected domains and experience levels. We recruited from three domains: Earth and climate science, the social sciences, and data journalism. Therefore, our findings may not generalize to geospatial data users outside these areas, such as epidemiologists, digital humanities researchers, or statisticians. We believe there are also opportunities for additional research *within* our chosen domains and experience groups. The number of participants from any given subgroup of our participant pool is too small to reveal insights about the needs of each class independently.

In general, the sample size of our study ($n = 25$) limits our ability to make quantitative claims about the prevalence of our findings in a broader population [72]. Furthermore, qualitative research experts warn about the risks of quantifying qualitative data [25]. For these reasons, we intentionally avoided attempts to generalize our findings beyond our participant group. Instead, we hope they can serve as a basis for designing larger-scale studies to assess the prevalence of our identified challenges in the wider community of geospatial data users.

Using contextual inquiry with open task selection gave us rich detail on participants' challenges but also came with drawbacks. First, asking participants to narrate their thought processes while performing cognitively challenging tasks can make these tasks more difficult [24]. Thus, tasks may appear harder in a lab setting than in a non-observational context. We attempted to mitigate this effect by permitting participants to pause narration until they reached a

stopping point for discussion. Second, asking participants to work on their own tasks inhibits us from making comparative claims that a fixed-task design may support. For example, alternative studies could compare participant performance on fixed tasks across multiple tools to understand their relative strengths. Additionally, our study design does not assess whether participants' tasks are representative of the work of geospatial data users more generally. Validating task representativeness through additional studies would bolster our findings.

A critical next step for this research is to validate our design opportunities with domain experts. We followed the practice of other contextual inquiry studies that derive design opportunities directly from participant observation [11, 19, 53]. Indeed, a strength of contextual inquiry is that it exposes unforeseen participant challenges in situ, allowing us to identify needs that may not become visible in alternative methods that rely on participants' memories of their work. However, this does not erase the threats of researcher confirmation bias [73] or causal error [76]. Future work can test our design opportunities through expert interviews, large-scale surveys, and formative studies of new systems.

8 CONCLUSION

As geospatial data grows in scale and accessibility, domain experts require increasingly expressive tools to harness the insights hidden in this data. But what could such tools look like, and what challenges should they address? This study deepens our understanding of the computing needs of domain expert geospatial data users. Using contextual inquiry, we identified unreported challenges across five phases of participants' work: data discovery, data transformation, analysis, analysis representation, and visualization. For example, our work is the first to discuss how users (1) employ data subsetting and resolution reduction to speed up exploratory geospatial analysis, (2) create informal program representations to record geoprocessing workflows, and (3) observe changes to feature counts and geometry to infer geospatial operator behavior. Going beyond prior work on GIS usability, we also uncovered needs that extend to other tools used in modern geospatial workflows, including computational notebooks, design software, and geospatial analysis and visualization libraries. Our observations revealed that four challenges—finding and transforming geospatial data to satisfy spatiotemporal constraints, understanding the behavior of geospatial operators, tracking geospatial data provenance, and exploring the cartographic design space—were especially difficult for participants. From these observations, we synthesized novel design opportunities for geospatial analysis and visualization systems. Future work can build on these insights to create useful and usable tooling that makes it easier to explore, analyze, and communicate patterns of spatiotemporal change in our environment and societies.

ACKNOWLEDGMENTS

We are indebted to our anonymous participants for sharing their working practices with geospatial data—this research would not have been possible without them. We are also grateful to our anonymous reviewers for their thoughtful and incisive comments. Finally, we would like to thank Rachel Leven at the Center for Computing, Data Science, and Society at the University of California, Berkeley for connecting us with data journalists for this study. This work is supported in part by NSF grants FW-HTF 2129008, CA-HDR 1936731, and CA-HDR 2033558, as well as by gifts from Google, Microsoft, and Sigma Computing. Sarah E. Chasins is a Chan Zuckerberg Biohub Investigator.

REFERENCES

[1] QGIS Association. 2022. 24.4. The history manager — QGIS Documentation. https://docs.qgis.org/3.22/en/docs/user_manual/processing/history.html. Accessed: 2022-06-24.

[2] QGIS Association. 2022. 24.5. The graphical modeler — QGIS Documentation. https://docs.qgis.org/3.22/en/docs/user_manual/processing/modeler.html. Accessed: 2022-06-27.

[3] QGIS Association. 2022. 25.1.17. Vector general — QGIS Documentation. https://docs.qgis.org/3.22/en/docs/user_manual/processing_algs/qgis/vectorgeneral.html. Accessed: 2022-06-21.

[4] QGIS Association. 2022. QGIS Geographic Information System. https://qgis.org/. Accessed: 2022-08-22.

[5] Marie-Aude Aufaure-Portier. 1995. Definition of a Visual Language for GIS. In *Cognitive Aspects of Human-Computer Interaction for Geographic Information Systems*, Timothy L. Nyerges, David M. Mark, Robert Laurini, and Max J. Egenhofer (Eds.). Springer Netherlands, Dordrecht, Netherlands, 163–178. https://doi.org/10.1007/978-94-011-0103-5_12

[6] Shaon Barman, Sarah Chasins, Rastislav Bodik, and Sumit Gulwani. 2016. Ringer: Web Automation by Demonstration. *ACM SIGPLAN Notices* 51, 10 (Oct. 2016), 748–764. https://doi.org/10.1145/3022671.2984020

[7] Rohan Bavishi, Caroline Lemieux, Roy Fox, Koushik Sen, and Ion Stoica. 2019. AutoPandas: Neural-Backed Generators for Program Synthesis. *Proceedings of the ACM on Programming Languages* 3, OOPSLA (Oct. 2019), 168:1–168:27. https://doi.org/10.1145/3360594

[8] Alan F. Blackwell and Thomas R.G. Green. 2003. Notational Systems — the Cognitive Dimensions of Notations Framework. In *HCI Models, Theories, and Frameworks: Toward a Multidisciplinary Science*, John M. Carroll (Ed.). Morgan Kaufmann, San Francisco, CA, USA, Chapter 5, 103–133. https://doi.org/10.1016/B978-155860808-5/50005-8

[9] Matthew Bloch. 2022. Mapshaper. https://mapshaper.org/. Accessed: 2022-07-13.

[10] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative Research in Psychology* 3, 2 (April 2006), 77–101. https://doi.org/10.1191/1478088706qp063oa

[11] Julia Brich, Marcel Walch, Michael Rietzler, Michael Weber, and Florian Schaub. 2017. Exploring End User Programming Needs in Home Automation. *ACM Transactions on Computer-Human Interaction* 24, 2 (April 2017), 11:1–11:35. https://doi.org/10.1145/3057858

[12] U.S. Census Bureau. 2020. American Community Survey 5-Year Estimates. S2502: Demographic Characteristics for Occupied Housing Units. https://data.census.gov/. Retrieved from: https://data.census.gov/table?t=Owner/Renter+(Householder)+Characteristics&g=0100000US%240500000&tid=ACSST5Y2020.S2502. Type: dataset.

[13] Brian Burg, Richard Bailey, Amy J. Ko, and Michael D. Ernst. 2013. Interactive Record/Replay for Web Application Debugging. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology (UIST '13)*. Association for Computing Machinery, New York, NY, USA, 473–484. https://doi.org/10.1145/2501988.2502050

[14] Kang-Tsung Chang. 2019. *Geographic Information System*. John Wiley & Sons, Ltd, Hoboken, NJ, USA, 1–10. https://doi.org/10.1002/9781118786352.wbieg0152.pub2

[15] Sarah Chasins, Shaon Barman, Rastislav Bodik, and Sumit Gulwani. 2015. Browser Record and Replay as a Building Block for End-User Web Automation Tools. In *Proceedings of the 24th International Conference on World Wide Web (WWW '15 Companion)*. Association for Computing Machinery, New York, NY, USA, 179–182. https://doi.org/10.1145/2740908.2742849

[16] Sarah E. Chasins, Maria Mueller, and Rastislav Bodik. 2018. Rousillon: Scraping Distributed Hierarchical Web Data. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology (UIST '18)*. Association for Computing Machinery, New York, NY, USA, 963–975. https://doi.org/10.1145/3242587.3242661

[17] Luca Cinquini, Daniel Crichton, Chris Mattmann, John Harney, Galen Shipman, Feiyi Wang, Rachana Ananthakrishnan, Neill Miller, Sebastian Denvil, Mark Morgan, Zed Pobre, Gavin M. Bell, Charles Doutriaux, Robert Drach, Dean Williams, Philip Kershaw, Stephen Pascoe, Estanislao Gonzalez, Sandro Fiore, and Roland Schweitzer. 2014. The Earth System Grid Federation: An open infrastructure for access to distributed geospatial data. *Future Generation Computer Systems* 36 (July 2014), 400–417. https://doi.org/10.1016/j.future.2013.07.002

[18] Open Geospatial Consortium. 2022. GeoSPARQL. https://opengeospatial.github.io/ogc-geosparql/geosparql11/spec.html#_topology_vocabulary_extension. Accessed: 2022-12-12.

[19] Bronwyn J. Cumbo, Tom Bartindale, and Dan Richardson. 2021. Exploring the Opportunities for Online Learning Platforms to Support the Emergency Home School Context. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. Association for Computing Machinery, New York, NY, USA, 1–11. https://doi.org/10.1145/3411764.3445044

[20] Christopher Daly, Ronald P. Neilson, and Donald L. Phillips. 1994. A Statistical-Topographic Model for Mapping Climatological Precipitation over Mountainous Terrain. *Journal of Applied Meteorology and Climatology* 33, 2 (Feb. 1994), 140–158. https://doi.org/10.1175/1520-0450(1994)033<0140:ASTMFM>2.0.CO;2 Type: dataset.

[21] Clare Davies and David Medyckyj-Scott. 1994. GIS usability: recommendations based on the user's view. *International Journal of Geographical Information Science* 8, 2 (1994), 175–189. https://doi.org/10.1080/02693799408901993

[22] Clare Davies and David Medyckyj-Scott. 1995. Feet on the Ground: Studying User-GIS Interaction in the Workplace. In *Cognitive Aspects of Human-Computer Interaction for Geographic Information Systems*, Timothy L. Nyerges, David M. Mark, Robert Laurini, and Max J. Egenhofer (Eds.). Springer Netherlands, Dordrecht, Netherlands, 123–141. https://doi.org/10.1007/978-94-011-0103-5_10

[23] Clare Davies and David Medyckyj-Scott. 1996. GIS users observed. *International Journal of Geographical Information Systems* 10, 4 (June 1996), 363–384. https://doi.org/10.1080/02693799608902085

[24] Simon P. Davies and Adrian M. Castell. 1994. From Individuals to Groups Through Artifacts: The Changing Semantics of Design in Software Development. In *User-Centred Requirements for Software Engineering Environments*, David J. Gilmore, Russel L. Winder, and Françoise Détienne (Eds.). Springer-Verlag, New York, NY, USA, 11–23.

[25] Norma K. Denzin and Yvonna S. Lincoln (Eds.). 2018. *The SAGE Handbook of Qualitative Research* (5th ed.). SAGE Publications, Inc., Thousand Oaks, CA, USA.

[26] Ian Drosos, Titus Barik, Philip J. Guo, Robert DeLine, and Sumit Gulwani. 2020. Wrex: A Unified Programming-by-Example Interaction for Synthesizing Readable Code for Data Scientists. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3313831.3376442

[27] M. Drusch, U. Del Bello, S. Carlier, O. Colin, V. Fernandez, F. Gascon, B. Hoersch, C. Isola, P. Laberinti, P. Martimort, A. Meygret, F. Spoto, O. Sy, F. Marchese, and P. Bargellini. 2012. Sentinel-2: ESA's Optical High-Resolution Mission for GMES Operational Services. *Remote Sensing of Environment* 120 (May 2012), 25–36. https://doi.org/10.1016/j.rse.2011.11.026

[28] Max Egenhofer. 1995. User Interfaces: Front Matter. In *Cognitive Aspects of Human-Computer Interaction for Geographic Information Systems*, Timothy L. Nyerges, David M. Mark, Robert Laurini, and Max J. Egenhofer (Eds.). Springer Netherlands, Dordrecht, Netherlands, 143–145.

[29] Miguel Elias, Jeremy Elson, Danyel Fisher, and Jon Howell. 2008. "Do I Live in a Flood Basin?" Synthesizing Ten Thousand Maps. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*. Association for Computing Machinery, New York, NY, USA, 255–264. https://doi.org/10.1145/1357054.1357100

[30] Esri. 2022. ArcGIS. https://www.esri.com/en-us/arcgis/about-arcgis/overview. Accessed: 2022-08-22.

[31] Esri. 2022. A complete listing of the Spatial Analyst tools—ArcGIS Pro Documentation. https://pro.arcgis.com/en/pro-app/2.8/tool-reference/spatial-analyst/complete-listing-of-spatial-analyst-tools.htm. Accessed: 2022-05-13.

[32] Esri. 2022. Merge (Data Management)—ArcGIS Pro Documentation. https://pro.arcgis.com/en/pro-app/2.8/tool-reference/data-management/merge.htm. Accessed: 2022-06-21.

[33] Esri. 2022. Viewing tool execution history—ArcMap Documentation. https://desktop.arcgis.com/en/arcmap/latest/analyze/executing-tools/history-log-files.htm. Accessed: 2022-06-24.

[34] Munazza Fatima, Kara J. O'Keefe, Wenjia Wei, Sana Arshad, and Oliver Gruebner. 2021. Geospatial Analysis of COVID-19: A Scoping Review. *International Journal of Environmental Research and Public Health* 18, 5 (Jan. 2021), 2336. https://doi.org/10.3390/ijerph18052336

[35] Thore Fechner, Dennis Wilhelm, and Christian Kray. 2015. Ethermap — Real-time Collaborative Map Editing. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. Association for Computing Machinery, New York, NY, USA, 3583–3592. https://doi.org/10.1145/2702123.2702536

[36] Melanie Feinberg. 2017. A Design Perspective on Data. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. Association for Computing Machinery, New York, NY, USA, 2952–2963. https://doi.org/10.1145/3025453.3025837

[37] G. David Garson and Robert S. Biggs. 1992. *Analytic Mapping and Geographic Databases*. SAGE Publications, Inc., Newbury Park, CA, USA. https://doi.org/10.4135/9781412983334

[38] GeoPandas. 2022. GeoPandas 0.12.2. https://geopandas.org/en/stable/. Accessed: 2023-01-06.

[39] GeoPandas. 2022. Merging Data — GeoPandas 0.12.2. https://geopandas.org/en/stable/docs/user_guide/mergingdata.html#attribute-joins. Accessed: 2023-01-06.

[40] Jianya Gong, Jing Geng, and Zeqiang Chen. 2015. Real-time GIS data model and sensor web service platform for environmental data management. *International Journal of Health Geographics* 14, 1 (Jan. 2015), 2. https://doi.org/10.1186/1476-072X-14-2

[41] Noel Gorelick, Matt Hancher, Mike Dixon, Simon Ilyushchenko, David Thau, and Rebecca Moore. 2017. Google Earth Engine: Planetary-scale geospatial analysis for everyone. *Remote Sensing of Environment* 202 (Dec. 2017), 18–27. https://doi.org/10.1016/j.rse.2017.06.031

[42] Philip J. Guo, Sean Kandel, Joseph M. Hellerstein, and Jeffrey Heer. 2011. Proactive Wrangling: Mixed-Initiative End-User Programming of Data Transformation Scripts. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. Association for Computing Machinery, New York, NY, USA, 65–74. https://doi.org/10.1145/2047196.2047205

[43] Mordechai (Muki) Haklay and Artemis Skarlatidou. 2010. Human-computer interaction and geospatial technologies – context. In *Interacting with Geospatial Technologies*. John Wiley & Sons, Ltd, West Sussex, UK, Chapter 1, 1–18. https://doi.org/10.1002/9780470689813.ch1

[44] Mordechai (Muki) Haklay and Antigoni Zafiri. 2008. Usability Engineering for GIS: Learning from a Screenshot. *The Cartographic Journal* 45, 2 (May 2008), 87–97. https://doi.org/10.1179/174327708X305085

[45] HashiCorp. 2023. Terraform. https://www.terraform.io/. Accessed: 01-07-2023.

[46] Brent Hecht, Johannes Schöning, Thomas Erickson, and Reid Priedhorsky. 2011. Geographic Human-Computer Interaction. In *CHI '11 Extended Abstracts on Human Factors in Computing Systems (CHI EA '11)*. Association for Computing Machinery, New York, NY, USA, 447–450. https://doi.org/10.1145/1979742.1979532

[47] Brent Hecht, Johannes Schöning, Muki Haklay, Licia Capra, Afra J. Mashhadi, Loren Terveen, and Mei-Po Kwan. 2013. Geographic Human-Computer Interaction. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems (CHI EA '13)*. Association for Computing Machinery, New York, NY, USA, 3163–3166. https://doi.org/10.1145/2468356.2479637

[48] Jeffrey Heer and Michael Bostock. 2010. Declarative Language Design for Interactive Visualization. *IEEE Transactions on Visualization and Computer Graphics* 16, 6 (nov 2010), 1149–1156. https://doi.org/10.1109/TVCG.2010.144

[49] Christian Heipke. 2010. Crowdsourcing geospatial data. *ISPRS Journal of Photogrammetry and Remote Sensing* 65, 6 (Nov. 2010), 550–557. https://doi.org/10.1016/j.isprsjprs.2010.06.005

[50] Brian Hempel, Justin Lubin, and Ravi Chugh. 2019. Sketch-n-Sketch: Output-Directed Programming for SVG. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology (UIST '19)*. Association for Computing Machinery, New York, NY, USA, 281–292. https://doi.org/10.1145/3332165.3347925

[51] Karen Holtzblatt and Hugh Beyer. 2017. Principles of Contextual Inquiry. In *Contextual Design: Design for Life* (2nd ed.), Karen Holtzblatt and Hugh Beyer (Eds.). Morgan Kaufmann Publishers, Boston, MA, USA, Chapter 3, 43–80. https://doi.org/10.1016/B978-0-12-800894-2.00003-X

[52] Sean Kandel, Andreas Paepcke, Joseph Hellerstein, and Jeffrey Heer. 2011. Wrangler: Interactive Visual Specification of Data Transformation Scripts. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. Association for Computing Machinery, New York, NY, USA, 3363–3372. https://doi.org/10.1145/1978942.1979444

[53] Anna Kawakami, Venkatesh Sivaraman, Hao-Fei Cheng, Logan Stapleton, Yanghuidi Cheng, Diana Qing, Adam Perer, Zhiwei Steven Wu, Haiyi Zhu, and Kenneth Holstein. 2022. Improving Human-AI Partnerships in Child Welfare: Understanding Worker Practices, Challenges, and Desires for Algorithmic Decision Support. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22)*. Association for Computing Machinery, New York, NY, USA, 1–18. https://doi.org/10.1145/3491102.3517439

[54] Mary Beth Kery, Amber Horvath, and Brad Myers. 2017. Variolite: Supporting Exploratory Programming by Data Scientists. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. Association for Computing Machinery, New York, NY, USA, 1265–1276. https://doi.org/10.1145/3025453.3025626

[55] Mary Beth Kery, Bonnie E. John, Patrick O'Flaherty, Amber Horvath, and Brad A. Myers. 2019. Towards Effective Foraging by Data Scientists to Find Past Analysis Choices. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3290605.3300322

[56] Mary Beth Kery, Marissa Radensky, Mahima Arya, Bonnie E. John, and Brad A. Myers. 2018. The Story in the Notebook: Exploratory Data Science using a Literate Programming Tool. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–11. https://doi.org/10.1145/3173574.3173748

[57] Laura Koesten, Emilia Kacprzak, Jeni Tennison, and Elena Simperl. 2019. Collaborative Practices with Structured Data: Do Tools Support What Users Need?. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3290605.3300330

[58] Manolis Koubarakis, Manos Karpathiotakis, Kostis Kyzirakos, Charalampos Nikolaou, and Michael Sioutis. 2012. Data Models and Query Languages for Linked Geospatial Data. In *Reasoning Web: Semantic Technologies for Advanced Query Answering*, Thomas Eiter and Thomas Krennwallner (Eds.). Springer-Verlag, Berlin and Heidelberg, Germany, 290–328. https://doi.org/10.1007/978-3-642-33158-9_8

[59] M. J. Kraak. 2021. *Cartography: Visualization of Geospatial Data* (4th ed.). Routledge, Boca Raton, FL, USA. https://doi.org/10.1201/9780429464195

[60] Nicolas Lambert. 2022. Bertin.js. https://github.com/neocarto/bertin. Accessed: 2022-12-14.

[61] Jae-Gil Lee and Minseo Kang. 2015. Geospatial Big Data: Challenges and Opportunities. *Big Data Research* 2, 2 (June 2015), 74–81. https://doi.org/10.1016/j.bdr.2015.01.003

[62] Sorin Lerner. 2020. Projection Boxes: On-the-fly Reconfigurable Visualization for Live Programming. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–7. https://doi.org/10.1145/3313831.3376494

[63] Gilly Leshed, Eben M. Haber, Tara Matthews, and Tessa Lau. 2008. CoScripter: Automating & Sharing How-To Knowledge in the Enterprise. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*. Association for Computing Machinery, New York, NY, USA, 1719–1728. https://doi.org/10.1145/1357054.1357323

[64] María-Jesús Lobo, Emmanuel Pietriga, and Caroline Appert. 2015. An Evaluation of Interactive Map Comparison Techniques. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. Association for Computing Machinery, New York, NY, USA, 3573–3582. https://doi.org/10.1145/2702123.2702130

[65] Steven M. Manson, Len Kne, Kevin R. Dyke, Jerry Shannon, and Sami Eria. 2012. Using Eye-tracking and Mouse Metrics to Test Usability of Web Mapping Navigation. *Cartography and Geographic Information Science* 39, 1 (2012), 48–60.

[66] Jon May and Tim Gamble. 2014. Collocating Interface Objects: Zooming into Maps. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. Association for Computing Machinery, New York, NY, USA, 2085–2094. https://doi.org/10.1145/2556288.2557279

[67] James Mickens, Jeremy Elson, and Jon Howell. 2010. Mugshot: Deterministic Capture and Replay for Javascript Applications. In *Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation (NSDI '10)*. USENIX Association, USA, 11.

[68] Microsoft. 2022. Microsoft Planetary Computer. https://planetarycomputer.microsoft.com/. Accessed: 2022-09-10.

[69] Anders Miltner, Sumit Gulwani, Vu Le, Alan Leung, Arjun Radhakrishna, Gustavo Soares, Ashish Tiwari, and Abhishek Udupa. 2019. On the Fly Synthesis of Edit Suggestions. *Proceedings of the ACM on Programming Languages* 3, OOPSLA (Oct. 2019), 143:1–143:29. https://doi.org/10.1145/3360569

[70] Dominik Moritz, Chenglong Wang, Greg L. Nelson, Halden Lin, Adam M. Smith, Bill Howe, and Jeffrey Heer. 2019. Formalizing Visualization Design Knowledge as Constraints: Actionable and Extensible Models in Draco. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (Jan. 2019), 438–448. https://doi.org/10.1109/TVCG.2018.2865240

[71] Michael Muller, Ingrid Lange, Dakuo Wang, David Piorkowski, Jason Tsay, Q. Vera Liao, Casey Dugan, and Thomas Erickson. 2019. How Data Science Workers Work with Data: Discovery, Capture, Curation, Design, Creation. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–15. https://doi.org/10.1145/3290605.3300356

[72] Brad A. Myers, Andrew J. Ko, Thomas D. LaToza, and YoungSeok Yoon. 2016. Programmers Are Users Too: Human-Centered Methods for Improving Programming Tools. *Computer* 49, 7 (July 2016), 44–52. https://doi.org/10.1109/MC.2016.200

[73] Raymond S. Nickerson. 1998. Confirmation Bias: A Ubiquitous Phenomenon in Many Guises. *Review of General Psychology* 2, 2 (June 1998), 175–220. https://doi.org/10.1037/1089-2680.2.2.175

[74] Observable. 2022. Observable. https://observablehq.com/. Accessed: 2022-09-11.

[75] Observable. 2022. Observable Plot. https://observablehq.com/plot. Accessed: 2022-12-14.

[76] Anthony J. Onwuegbuzie and Nancy L. Leech. 2007. Validity and Qualitative Research: An Oxymoron? *Quality & Quantity* 41, 2 (April 2007), 233–249. https://doi.org/10.1007/s11135-006-9000-3

[77] Jason Ott, Tyson Loveless, Chris Curtis, Mohsen Lesani, and Philip Brisk. 2018. BioScript: Programming Safe Chemistry on Laboratories-on-a-Chip. *Proceedings of the ACM on Programming Languages* 2, OOPSLA (Oct. 2018), 128:1–128:31. https://doi.org/10.1145/3276498

[78] Leysia Palen, Robert Soden, T. Jennings Anderson, and Mario Barrenechea. 2015. Success & Scale in a Data-Producing Organization: The Socio-Technical Evolution of OpenStreetMap in Response to Humanitarian Events. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. Association for Computing Machinery, New York, NY, USA, 4113–4122. https://doi.org/10.1145/2702123.2702294

[79] Fernando Perez and Brian E. Granger. 2007. IPython: A System for Interactive Scientific Computing. *Computing in Science & Engineering* 9, 3 (May 2007), 21–29. https://doi.org/10.1109/MCSE.2007.53

[80] Alenka Poplin. 2015. How user-friendly are online interactive maps? Survey based on experiments with heterogeneous users. *Cartography and Geographic Information Science* 42, 4 (Aug. 2015), 358–376. https://doi.org/10.1080/15230406.2014.991427

[81] The Selenium Project. 2022. Selenium. https://www.selenium.dev/. Accessed: 2022-12-06.

[82] Meredith P. Richards. 2014. The Gerrymandering of School Attendance Zones and the Segregation of Public Schools: A Geospatial Analysis. *American Educational Research Journal* 51, 6 (Dec. 2014), 1119–1157. https://doi.org/10.3102/0002831214553652

[83] Esther Rolf, Jonathan Proctor, Tamma Carleton, Ian Bolliger, Vaishaal Shankar, Miyabi Ishihara, Benjamin Recht, and Solomon Hsiang. 2021. A generalizable and accessible approach to machine learning with global satellite imagery. *Nature Communications* 12, 1 (July 2021), 4392. https://doi.org/10.1038/s41467-021-24638-z

[84] D. P. Roy, M. A. Wulder, T. R. Loveland, Woodcock C.E, R. G. Allen, M. C. Anderson, D. Helder, J. R. Irons, D. M. Johnson, R. Kennedy, T. A. Scambos, C. B. Schaaf, J. R. Schott, Y. Sheng, E. F. Vermote, A. S. Belward, R. Bindschadler, W. B. Cohen, F. Gao, J. D. Hipple, P. Hostert, J. Huntington, C. O. Justice, A. Kilic, V. Kovalskyy, Z. P. Lee, L. Lymburner, J. G. Masek, J. McCorkel, Y. Shuai, R. Trezza, J. Vogelmann, R. H. Wynne, and Z. Zhu. 2014. Landsat-8: Science and product vision for terrestrial global change research. *Remote Sensing of Environment* 145 (April 2014), 154–172. https://doi.org/10.1016/j.rse.2014.02.001

[85] RStudio. 2022. rmarkdown: Dynamic Documents for R. https://github.com/rstudio/rmarkdown R package version 2.19.2. Accessed: 2023-01-07.

[86] Steve Running, Qiaozhen Mu, and Maosheng Zhao. 2017. MYD16A2 MODIS/Aqua Net Evapotranspiration 8-Day L4 Global 500m SIN Grid V006. https://doi.org/10.5067/MODIS/MYD16A2.006 Type: dataset.

[87] Mark Santolucito, William T. Hallahan, and Ruzica Piskac. 2019. Live Programming By Example. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems (CHI EA '19)*. Association for Computing Machinery, New York, NY, USA, 1–4. https://doi.org/10.1145/3290607.3313266

[88] Arvind Satyanarayan, Dominik Moritz, Kanit Wongsuphasawat, and Jeffrey Heer. 2017. Vega-Lite: A Grammar of Interactive Graphics. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (jan 2017), 341–350. https://doi.org/10.1109/TVCG.2016.2599030

[89] VERBI Software. 2022. MAXQDA. https://maxqda.com/. Accessed: 2022-11-17.

[90] Kristin Stock and Hans Guesgen. 2016. Geospatial Reasoning With Open Data. In *Automating Open Source Intelligence*, Robert Layton and Paul A. Watters (Eds.). Syngress, Waltham, MA, USA, Chapter 10, 171–204. https://doi.org/10.1016/B978-0-12-802916-9.00010-5

[91] Carol Traynor and Marian G. Williams. 1995. Why Are Geographic Information Systems Hard to Use?. In *Conference Companion on Human Factors in Computing Systems (CHI '95)*. Association for Computing Machinery, New York, NY, USA, 288–289. https://doi.org/10.1145/223355.223678

[92] Carol Traynor and Marian G. Williams. 1997. A Study of End-User Programming for Geographic Information Systems. In *Papers presented at the seventh workshop on Empirical studies of programmers (ESP '97)*. Association for Computing Machinery, New York, NY, USA, 140–156. https://doi.org/10.1145/266399.266412

[93] Carol Traynor and Marian G. Williams. 2001. End Users and GIS: A Demonstration Is Worth a Thousand Words. In *Your Wish is My Command: Programming by Example*. Morgan Kaufmann Publishers, San Francisco, CA, USA, 115–134.

[94] René Unrau and Christian Kray. 2019. Usability evaluation for geographic information systems: a systematic literature review. *International Journal of Geographical Information Science* 33, 4 (April 2019), 645–665. https://doi.org/10.1080/13658816.2018.1554813

[95] René Unrau, Morin Ostkamp, and Christian Kray. 2017. An approach for harvesting, visualizing, and analyzing WebGIS sessions to identify usability issues. In *Proceedings of the ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS '17)*. Association for Computing Machinery, New York, NY, USA, 33–38. https://doi.org/10.1145/3102113.3102122

[96] USDA. 2022. USDA - National Agricultural Statistics Service - Research and Science - Cropland Data Layer Releases. https://www.nass.usda.gov/Research_and_Science/Cropland/Release/index.php. Accessed: 2022-07-12. Type: dataset.

[97] Leland Wilkinson. 2005. *The Grammar of Graphics* (2nd ed.). Springer Science+Business Media, Inc., New York, NY, USA. https://doi.org/10.1007/0-387-28695-0

[98] Chaowei Yang, Manzhu Yu, Fei Hu, Yongyao Jiang, and Yun Li. 2017. Utilizing Cloud Computing to address big geospatial data challenges. *Computers, Environment and Urban Systems* 61 (Jan. 2017), 120–128. https://doi.org/10.1016/j.compenvurbsys.2016.10.010

[99] Katherine Ye, Wode Ni, Max Krieger, Dor Ma'ayan, Jenna Wise, Jonathan Aldrich, Joshua Sunshine, and Keenan Crane. 2020. Penrose: From Mathematical Notation to Beautiful Diagrams. *ACM Transactions on Graphics* 39, 4 (July 2020), 144:144:1–144:144:16. https://doi.org/10.1145/3386569.3392375