Towards an Interactive, Geolocated Streamflow Visualization in the Browser

Brian Studwell
Human-Computer Interaction + Design
University of Washington
bwstudwell@gmail.com
(585)747-0404

ABSTRACT
In this paper I describe work towards a visualization joining geospatial data from the NHDPlus v2 hydrologic framework, with real-time streamflow values collected by the National Water Information System. GIS software is generally inaccessible to the public due to an expensive, proprietary license model and highly complex operation. Developing an approachable, in-browser experience for exploring this data will benefit water resource engineers in studying and responding to mounting concerns around the availability and quality of water, but also promises application in other fields. I survey existing works which have influenced my own, detail my approach for acquiring and processing data stored by multiple government services and discuss prospects for future work, including scaling these techniques to the national level and relating the visualization to other political, natural and industrial data of interest.

INTRODUCTION
This work was prompted by one of 2015's NASA Space Apps Challenges calling for a visualization of streamflow over time on a nationwide GIS layer of stream and river networks. The NHDPlus dataset is a richly detailed, comprehensive catalogue of water bodies in the contiguous United States, Hawaii and Puerto Rico. It captures a wide array of attributes beyond simple geometry and is meticulously maintained by a partnership between Horizon Systems and the U.S. Geologic Survey. The stream gages which serve the NWIS cannot match the scope of the NHDPlus, but measure values such as discharge, humidity and temperature, bringing the critical ideas of, "how much?", and, "when", to complement the well established, "where?" for this system.

Goals
As part of larger open data initiatives both of these datasets are available for public consumption and use. The hydrologic data, however, is in a format which does not lend itself to simple, web-based interaction, and streamflow is currently presented by the USGS in a visualization which cannot be zoomed and offers no ability to easily change the time period being viewed. Examining streamflow trends over time, in their geospatial contexts, at local, regional and national levels will reveal patterns and relationships which could not be communicated by space or time alone. Streamflow measurements could be correlated to seasonality, land-use, political events, weather phenomena and more. In its most straightforward application, the visualization would serve the needs of water resource managers in quickly evaluating and comparing trends. I imagine, however, use cases in a variety of other practices including climate science, agriculture, urban planning, activism, journalism, cartography and others.

Additionally, I hope that a visualization of this kind would add to the popular dialogue about water use in our society. Water consumption is a vitally important concern for America, and for the world at large. It is, however, a challenging concept to consider beyond one's personal frame of reference. Producing a compelling visualization which can be accessed by and communicate to members of the public will help raise awareness of water conservation issues by helping us understand, in immediate, visceral terms, just how numerous, widespread, and variable our water resources are.

RELATED WORK
I drew inspiration from a collection of works visualizing fluid and motion. I looked to projects which visualized GIS data in a richly detailed manner, which addressed the NHDPlus dataset directly, and which visualized the patterns of wind or water movement across large areas and over time. The wind map by Wattenberg and Viegas was a highly influential on my thinking, as was a related project visualizing streamflow by Ibrahim Demir at the University of Iowa, and the Earth and Tokyo wind maps by Cameron Beccario. I also learned a great deal from Nelson Minar's work producing a vector river map, which was, in turn, reinterpreted by Mike Bostock in his U.S. Rivers piece.

Interactive Maps

The Wind Map, first released in 2012, has enjoyed great popular success, translating a complex natural phenomena into a visual representation which viewers can immediately grasp. It was for this reason that I chose to study it. The piece employs comet-like tails to indicate wind speed and direction. These tails taper towards their end and animate along their path. The effect is directional, immediate, and mesmerizing.

Ibrahim Demir, a faculty researcher at the University of Iowa, produced a beautiful visualization of streamflow in October of 2014 that borrows heavily from the Wind Map's visual language. Water bodies appear as white, comet-like tails moving along their respective courses, and join larger, "trunk", clusters of waterway. The impression of motion is effective and clear. Demir chooses to lend larger stream and river clusters halos, which multiply as they converge giving a sense of width to large water bodies as compared to their headwaters. One unclear aspect of this piece, however, is how closely tied it is to actual geospatial information. While it appears to follow the paths of rivers in a broad sense, it shows a density of animated paths which suggests the author is interpolating in some way to cover map area. As the source code is unavailable, this remains to be studied.

Vector Maps

In Nelson Minar's Vector River Map from 2013 the author gives a detailed account of rendering a web map with vector image tiles using the NHDPlus dataset. This is one of the most well documented vector tile projects I could find, and was helpful in considering how to parse and serve the NHDPlus data. Minar's map depicts a filtered selection of all the streams and rivers in the U.S. with the ability to pan and zoom fluidly. There is no element of time, however. This work differentiates the respective size of waterbodies using line-width which, while somewhat effective, lacks the pleasing aesthetics and liquid visual metaphor of the comet trails from the projects previously mentioned.

Mike Bostock produced his own version of the Vector River Map shortly after Minar, as a static, highly detailed rendering. Rostock's project gives us the added benefit of TopoJSON, a tool for greatly reducing the size of GeoJSON data. While the piece in question was not interactive, the sheer volume of detail presented was an achievement, and demonstrated the feasibility of rendering the geospatial data for the entire hydrology in using web technologies, particularly the HTML Canvas.

WebGL and Mapping Large Data

As performance became a prominent challenge over the course of this project I was prompted to evaluate technological alternatives to SVG and, as a result, was exposed to the work of Eric Fisher at Mapbox, an open source mapping company based in Washington, D.C. Fisher and others focus on rendering extremely large GeoJSON datasets, as in his project, "6 billion tweets". WebGL promises significant performance increases over SVG in certain situations and Fisher's work on MapboxGL, the Mapbox javascript library for WebGL, and Tippecanoe, a utility for creating vector tiles from large GeoJSON data, was both inspirational and instructional. While I was unable to incorporate MapboxGL into this work, it is my impression that an effective visualization of the data with fast performance and rich detail will demand the GPU-added performance added by WebGL.

METHODS

The vast majority of my time in this project was spent attempting to acquire, process and reason about the data I am attempting to use. The two datasets are hosted by separate organizations, in a variety of formats, with multiple methods for retrieval, as multiple sets and subsets, with dense, technical documentation when any is available at all. I set out in this work to challenge myself; to grapple with a complex data project wherein I would be forced to learn how to find the data, how to access the data, how to inspect the data, how to manipulate the data and how to store and serve the data in my eventual product. In that regard this effort, which was a failure in terms of visualization, has been a tremendous success.

Data Acquisition

To realize the goal of my project I needed to acquire representations of the location and geometry for every waterway in the U.S., and a collection of values measured by stream gage, along with their respective locations. The NASA challenge description listed the NHDPlus and NWIS as data sources, so I began to explore them.

NHDPlus V2

NHDPlus describes itself as, "a geo-spatial, hydrologic framework dataset envisioned by the US Environmental Protection Agency". First released in 2006, the NHDPlus is now in version 2, initially published in 2012. It selects and combines features from the National Hydrography Dataset (NHD), the National Elevation Dataset (NED), and the Watershed Boundary Dataset (WBD). It divides the water resources of the United States according to a system of Hydrologic Unit Codes, which follow a four-tier schema of region, sub-region, accounting unit, and cataloging unit. The U.S. is divided into 21 major Hydrologic Regions, each of which contains either the drainage area of a major river, or the combined drainage area of several rivers.

NHDPlus data is available for download by region. The Pacific Northwest, for instance, including Washington, Oregon, Idaho, and parts of Montana and Northern California, is available for download as HUC 17. Within each region, the information for the region is available via direct http download or ftp as .7z archives. 7zip is an open-source file compression tool available for all operating systems. There are multiple files available for a given region. HUC 17 offers users 32 different files for download, not including the release notes pdf. Files are often grouped by subregion, with one group available for the whole.

NWIS

Where the NHDPlus data was relatively straightforward to acquire, the streamflow measurements were anything but. Collecting data from over 1.5 million sites, the USGS investigates the occurrence, quantity, quality, distribution, and movement of surface and underground waters nationwide. They offer a REST API with access to Instantaneous Values (values from a specific moment in time) dating back to October 1, 2007, site information, daily values, water quality and level services, and statistics services. With each of these options one can use a url generator to build a query which retrieves the pertinent data. The difficulty, however, lies in navigating the various parameters that can be fed to the URL.

Having never accessed a restful web service before, my first query attempted to download all data for all sites in the Northeast for a time period of 8 years. This approach was ineffective. Through study, trial and error, and email conversation with some extremely patient staff at the USGS I was able to start making more controlled requests.

Instantaneous values are measurements taken at a specific point in time. The gages providing this type of data allege to record measurements at 15 minute intervals. Daily values report average measurements for a given day, and are available for a time period extending back much further than the instantaneous system. The time period can be set as a period from now (e.g. go back 4 days, 8 hours from now) or as a a range with start and stop date times. Of particular importance to this work was understanding, “parameter codes”. Parameter codes are 5 digit values passed as an optional argument in the URL which dictate the type of values to be retrieved. If none are included, all time series values for the sites of interest are retrieved, resulting in a great deal of unnecessary information. The parameter code 00060 requests streamflow values. All data is stored in Water ML, a technology standard based on XML for the encoding and exchange of hydrologic time series data. The web service, however, allows retrieval of the data in JSON.

Data Exploration

QGIS

After retrieving the NHDPlus shapeless I wanted to look at them and see what I was working with. QGIS is the recommended tool for handling shapefiles but, as a note of caution, open source does not mean simple. QGIS is a powerful and complex application with a steep learning curve. Another lesson I learned the hard way was that shapefiles are dependent on a suite of other files packaged with them, including .dbf and projection information. If separated, the shapefiles will not render. Once I figured it out QGIS gave me a way to easily evaluate the contents of my shapefiles and consider possible designs for my visualization.

D3

Mapping with D3 is no small task. It requires getting comfortable with various geospatial data tools, Geo and TopoJSON formats, the whacky world of map projections, and bringing that all into various coordinate spaces. I met limited success using D3 to inspect the flowline data from the NHDPlus. When I did get something to render in the browser it often caused the browser page to crash.

What D3 was particularly useful for was actually understanding what was in my data. Early in my exploration phase I had an enormous JSON file full of the geometries I wanted to analyze. I could not, however, open it. Trying to open the file in my text editor, Sublime Text, took several minutes to load. D3’s built in JSON reader allowed me to load the data into the browser and interactively view it in the console, which, in turn, meant I could determine how to access structures further down in the JSON hierarchy for subsequent D3 and Python code.

Tableau

I would have liked to use Tableau more in the early stages of this project, but the data coming from its source was not in a format Tableau could readily deal with. In the later stages of work as I was trying to understand where in the larger regions the stream gage sites I had retrieved were located I was able to add their information to a Tableau worksheet and very easily see the coverage, and lack thereof, of stream gages across the Northwest, which had strong implications for what the visualization I hoped to achieve might look like as there were significant gaps.

Data Processing

Shapefiles

Spatial information from the NHDPlus is stored in shapefiles, a vector data format which encodes geometric features as combinations of points, lines and polygons embedded with attributes. Because I began this work as a D3 project, I turned first to D3 mapping tutorials, and began learning ogr2ogr, a command-line spatial data conversion tool that exists as part of the larger Geospatial Data Abstraction Library. First attempts at using ogr2ogr were a challenge, as I was unfamiliar the UNIX conventions which underly many CLIs. Once learned, however, ogr2ogr makes it extremely easy to convert a shapefile to GeoJSON which could be processed by D3. Going one step further, GeoJSON can be converted into TopoJSON by Mike Bostock’s tool of the same name. TopoJSON is a more efficient encoding for geospatial data, resulting in decreased file size. One can write TopoJSON directly from a shapefile, which offers the advantage of supporting streaming. If the GeoJSON data you are working with is large, TopoJSON will refuse to process it.

If you instead process the shape file directly, TopoJSON can be coerced into working.

Parsing & Manipulating JSON

Some of my greatest challenges stemmed from attempting to programmatically acquire, parse, extract, and rebuild JSON data containing streamflow values. Because the USGS web service is restful, it is possible to simply enter a query URL directly into the browser and view the results. This is, however, slow, requires saving the data again in order to act upon it in any way, and labor intense. Another technology goal of mine in carrying out this work was to gain exposure to Python in the practice of gathering and cleaning data. I had no experience with the Python language prior, and developing a functioning script proved time consuming.

The information sent from NWIS adheres to the structure of WaterML, as mentioned above. In my use case, this results in a large number of extraneous and empty fields, as well as a needlessly complex information hierarchy. Because my goal was a browser-based visualization and I had already encountered problems with the size of the spatial data I resolved to retrieve only the information required for the project. All I needed was the name of the site, its latitude and longitude coordinates, and streamflow values at given times.

Using the Python Requests module I authored a script which would return a JSON response object from the NWIS. When working with JSON in the browser I could use the console to interactively explore its structure. I had to learn how to use the Python interpret to perform this role in my analysis, leaning heavily on .keys() and print() to see what was actually contained in various slices of my data. After lengthy trial and error I got the script to a point where it was reading the JSON response, extracting the values that I needed, and recombining them into a new, much smaller, valid (almost), file. Unfortunately, I was building the new JSON object from strings which, I was told by kindly strangers on Stackoverflow, is an ill advised, error-prone way of going about it.

Near the end of the time allowed for this work I still had not succeeded in debugging my script, and was forced to abandon it, having already invested disproportionate time in its development.

Vector Tiles

With the JSON previously produced I attempted to render the rivers and streams of the PNW in my browser. This was unsuccessful. At first I believed I was writing invalid D3 code, but after ensuring the code was not the problem I identified the culprit. The raw GeoJSON output from ogr2ogr weighed in at 438mb. Even after conversion to TopoJSON I was still over 400mb. This is due to the fact that Topojson derives much of its frugality from eliminating redundant borders. If a border is shared by two polygons it is only rendered once. The rivers and streams data, however, is an enormous collection of line segments, with no redundancies to prune. I needed to look for alternatives.

During initial research for the project I had come across the concept of tiles, an approach to web mapping wherein the map data to be rendered is divided into tiled regions, with sets of tiles being produced for a range of zoom levels. This allows the browser to render only the data necessary for whatever context is within view, caching the tiles for rapid retrieval and fluid transitions. Google Maps brought tile maps into the spotlight with maps that could pan and zoom fluidly, as opposed to the onerous reloading of traditional technologies. Vector tiles are a newer implementation of map tiles which, rather than producing sets of rasters which are stored and retrieved, embed the attributes of the data directly into the tile allowing for dynamic rendering in the browser. Vector tiling is the approach demonstrated in Minar's project referenced above. Because I hoped to develop an interactive visualization, this seemed like the right tool for the job.

There are a number of options out there to create tiles but in searching for a solution for particularly large datasets I came across a new utility called Tippecanoe, being developed by employees of Mapbox. Tippecanoe is a command line tool which can process arbitrarily large sets of GeoJSON data into map tiles, preserving data resolution across zoom levels, which seemed like exactly what I was trying to do. After a few halting first attempts I actually got it to work, and it produced a very nice .mbtiles formatted file containing flowlines for every river and stream in the Northwest. I did not, however, know what to do with that file. Mapbox offers a desktop editor called Mapbox Studio, but I was unable to load the .mbtiles file into Studio as a datasource. Running out of time, I looked for other ways to work with vector tiles.

PostGIS

PostgreSQL is a widely adopted open source database. PostGIS is an extension to PostgreSQL adding functionality specific to working with geospatial data. Several sources pointed to using PostGIS as the best option for handling my vector tiles, so I set out to learn SQL. Over time I was able to load my .mbtiles file into a PostGIS database. I was then able to use Mapbox Studio to view the flowlines for the first time. This visualization was still quite slow, as the geometries had not been optimized in any way, but at least they showed up.

Design

While no final designs have been completed at the time of this writing, I have considered several directions for what an implementation may look like. The basic structure is a larger overview of the region being viewed. This view will show flowlines for major water bodies as deemed appropriate for its level of zoom. Showing all of the rivers and streams at all zoom levels introduces problems of legibility and noticeable differences. I imagine the overview graphic will allow for a range of the highest level zooms, perhaps extending from viewing an area equivalent to several counties up to a view encompassing an entire HUC region.

The user will be provided a brush for dynamically selecting portions of the main graphic. The selected region will be reflected below in a secondary graphic which is, in effect, a loupe, or a magnified recreation of the selection. This loupe view will present higher detail and magnification, allowing close inspection of local features. The stream gage sites which fall within the selection will also be displayed in a line graph below the primary graphic which represents the period of record for the data. The default view will be 1 month, but I would like to allow the user to expand or contract this figure to their liking. In this way a viewer of the visualization may choose a region, two levels of zoom.

CONCLUSION

While I am ultimately disappointed in my failure to create an interactive visualization, I have learned a great deal about what it takes to get usable data in the first place. I would like to continue working on this project, as the subject matter is compelling and the potential for visualization to contribute meaningfully to the discourse around water use is high. Things which I knew nothing about going in to this project, which I now know slightly more than nothing about include, but are not limited to:

- Python
- Shapefiles and Mapping in D3
- GDAL, ogr2ogr, QGIS and other open geo tools
- JSON, GeoJSON and TopoJSON
- PostgreSQL, PostGIS, and SQL
- Vector Tiles
- Fundamental UNIX commands like cat, |, and echo

FUTURE WORK

The opportunities to build upon this work are numerous. Most obviously, of course, is the implementation of a working visualization. Assuming a small scale visualization is achieved, similar principles applied at the national level would be a priority.

I would like add additional means by which users could select data to display, including searching by zip code and place name, a clickable list of the largest rivers in the U.S., and list of pre-selected points of interest.

In the interest of usefulness it would be beneficial to include the ability to compare multiple views, perhaps as a, “split-screen”, or in small multiples. The ability to save views for later viewing would also be highly valuable.

I had hoped to explore multiple representations for the time series data beyond the simple line graph. Finding ways to highlight and visualize the time series within the map itself would be desirable, as would identifying meaningful scales and visualizations for subsets of the data (seasons, high/low, water-body type).

My goal for this project was to locate stream gages within the water body which they belonged to. The stream gages data includes a HUC number, which can be correlated to the HUC number of hydrologic features from the NHDPlus. The NHDPlus data also has means by which we can determine the larger structure that a given segment is a part of, including headwaters and tributaries. These include Strahler order, site ID, and GNIS_ID. If you target a selection of stream gages belonging to the same water body you can model the flow between them, creating many interesting possibilities for visualization at mid to high zoom levels.

Lastly, portraying this data alongside data from domains related to water use could provide for interesting comparison and debate. I found, and had hoped to include, records of municipal spending on water infrastructure, statistics of land-use across the nation, and historical records of major weather events, all of which could be laid over the streamflow data itself.

ACKNOWLEDGMENTS

I owe thanks to the employees who fielded my pleas for help at the NWIS support email account, members of the Mapbox support team for valiantly attempting to interpret my indecipherable questions, and various users of Stackoverflow and the GIS Stackexchange, who were both very helpful and vaguely insulting.

REFERENCES

1. *NHDPlus*
   Horizon Systems & The USGS
   http://www.horizon-systems.com/NHDPlus/index.php
2. *NWIS*
   http://waterdata.usgs.gov/nwis
1. *Wind Map*
   Martin Wattenberg and Fernanda Viegas
   http://hint.fm/projects/wind/
2. *Interactive Streamflow Visualization*
   Ibrahim Demir
   https://www.youtube.com/watch?v=ihtLeVxSBPA
3. *Earth and Tokyo Wind Maps*
   Cameron Beccario
   https://github.com/cambecc
4. *Vector River Map*
   Nelson Minar
   https://github.com/NelsonMinar/vector-river-map
5. *U.S. Rivers*
   Mike Bostock
   https://github.com/mbostock/us-rivers
6. *Tippecanoe*
   https://github.com/mapbox/tippecanoe