aQua: A Framework for Spatiotemporal Analysis and Visualizations of Water Quality Data at Scale

Matthew Young
Computer Science Department
Colorado State University
Fort Collins, Colorado, United States
asterix@rams.colostate.edu

Sangmi Pallickara
Computer Science Department
Colorado State University
Fort Collins, Colorado, United States
Sangmi.Pallickara@colostate.edu

Shrideep Pallickara
Computer Science Department
Colorado State University
Fort Collins, Colorado, United States
Shrideep.Pallickara@colostate.edu

Abstract—**Spatial data volumes have grown exponentially alongside the proliferation of sensing equipment and networked observational devices. In this study, we describe our framework aQua for performing visualizations and exploration of spatiotemporally evolving phenomena at scale. We validate our ideas in the context of data from the National Hydrology Database (NHD) and the Environmental Protection Agency (EPA) to support longitudinal analysis (53 years of data) for the vast majority of water bodies in the United States. Our methodology addresses issues relating to preserving interactivity, effective analysis, GPU accelerated visualizations, dynamic query generation, and scaling. We consider optimizations and refinements at the server-side, client-side, and how information exchange occurs between the client and server-side. We report both quantitative and qualitative assessments of several aspects of our tool to demonstrate its suitability. Finally, our methodology is broadly applicable to domains where visualization-driven explorations of spatiotemporally evolving phenomena are needed.**

Index Terms—**Spatiotemporal Phenomena, Big Data, Visualizations, and High Dimensional Analyses.**

I. Introduction

The past couple of decades have seen an exponential growth in the amount of data being generated [1]. This growth has been fueled in part by increases in the number of data sources, falling costs for sensing equipment, increased battery capacities, improvements in the quality and capacity of networks, and the proliferation of sensors that measure features with increased precision. Other data sources include simulations and models.

A substantial amount of data that are generated is spatial, i.e., data that have spatial coordinates associated with them. These data can either be point measurements with geotagged *[lat, long]* coordinates or N-sided polygons where each vertex is defined using a *[lat, long]* tuple. The data also have a timestamp representing when those observations were made. Data collections that are geocoded alongside timestamps are spatiotemporal datasets.

Spatiotemporal datasets arise naturally in several domains such as geosciences, urban planning, agriculture, environmental sciences, and ecology among others – where the phenomena are monitored using *in situ* or remote sensing equipment. These data provide opportunities to understand phenomena or inform decision making.

As spatiotemporal datasets proliferate, researchers are interested in tools that facilitate analyses over them [2]–[4]. In particular, researchers are interested in identifying the implicit characteristics associated with the spatiotemporal evolution of the feature space [5]–[7]. These include visualizations at vast spatiotemporal scales, spatial variation of the measure of interest and also how they vary over time for longitudinal analyses [8], [9]. However, researchers are stymied in their analyses by issues stemming from data volumes, the amount of I/O that needs to be performed, and precisely identifying the data of interest [10].

The overarching objective of this study is to leverage visualizations as a complement that undergirds complex data analysis over spatiotemporally evolving phenomena where both the spatial extent and time scales under consideration are vast. A key goal is to accomplish these visualization driven analyses and explorations while preserving interactivity [11].

The analysis and visualizations work in concert with each other. Visualization is one of the primary ways to explore both the raw data and the results of the analyses. These visualizations themselves inform subsequent analyses [12]. Queries identify spatiotemporal scopes where the specified predicates hold true and can be used to identify how the analyses proceed.

We validate our ideas in the context of water quality analyses. Our study involves every water body in the United States, every HUC12 watershed boundary, every census county and tract, and 1 million monitoring stations that profile and track over 5000 chemicals in these various spatial extents. The research questions and methodological aspects underpinning our work are broadly applicable to other spatiotemporally evolving phenomena such as urban sustainability and planning, assessing air quality, climatic clarifications, profiling environmental and ecological impacts of manmade and natural processes.

A. Research Challenges

Challenges in effectively supporting visualizations over spatiotemporally evolving phenomena include:

1) The number of visual elements: As the number of visual elements increases, cumulative overheads associated with visualization and analysis also increase.

2) Complexity and diversity of the visual elements: The visual elements may have a large number of vertices each with a double precision *[lat, long]* tuple associated with them. The complexity of these shapes increases the per-element rendering overheads.
3) Input/Output (I/O) overheads: Data fetching operations are I/O bound because they entail both disk I/O (retrievals) and network I/O (transmissions).
4) Variability of Features: Each visual element may have a large number of variable features associated with it.
5) The data are continually evolving with new measurements being reported continually. Furthermore no single spatial extent has all features being tracked.

B. Research Questions

The crux of this study is to enable interactive visualizations and analysis over voluminous, high-dimensional spatiotemporal data. Research questions that we explore include:
RQ-1: How can we preserve interactivity during explorations?
RQ-2: How can we limit the amount of I/O being performed?
RQ-3: How can we support rich data analyses at scale?

C. Approach Summary

Our methodology involves a carefully calibrated mix of (1) data preprocessing so that computationally expensive operations are not duplicated or performed in the critical path, (2) ensuring support for expressive analyses, (3) preserving interactivity, (4) coping with issues of scale, and (5) supporting exploratory analyses. We consider optimizations and refinements at the server-side, client-side, and information exchange between the client and server-side.

Our data preprocessing operations target creation of harmonized datasets, computing intermediate values needed for downstream analyses, and producing simplified representations of complex visual elements. Water bodies in our analyses are represented using shapefiles (N-sided polygons) from the NHD. Monitoring stations managed by the U.S. Environmental Protection Agency (EPA) report measurements for observed concentrations of diverse chemicals alongside the station's GPS coordinates and measurement timestamps. Our tool, aQua (URL for tool @ [13]), supports explorations over data reported by 1 million monitoring stations (for the 53-year duration of the data that we consider). Our data preprocessing targets associating measurements to spatial extents based on their proximity and inclusion within the shapefiles. Once the data sources, spatial extents, and measurements are harmonized the data analysis can proceed over spatial extents where individual data items are reported as multidimensional vectors with measurement timestamps.

To support effective analysis, we include support for tracking changes in feature values over time alongside support for drill-down and roll-up analyses. We supplement this with an emphasis on user experience that blends interactivity with simplicity of query composition, rich visualization, and graphing support to analyze spatiotemporally evolving phenomena. To effectively track and graph changes in feature values over time, we leverage discretization and binning to reduce the number of observations that need to be retrieved. Our application is designed as a Web Mercator projection system.

Query composition in aQua is dynamic and steered from the visualization engine. The queries are *declarative*; the details of the query composition and predicate formulation are shielded from the user. The framework generates these queries dynamically by constraining temporal bookends, spatial extents under consideration, and the order in which the query predicates are evaluated (based on features that have been indexed) – this allows pruning of the search spaces during query evaluations.

To ensure interactivity we rely on a mix of streaming, targeted rendering operations that account for boundary conditions, and shape simplifications. We reduce computational requirements by only rendering visual elements that exceed perceptual limits: visual elements that are excluded from rendering are included for rendering as a user zooms in.

Exploratory analyses are backed using dynamic query compositions alongside the configuration of powerful defaults for several aspects with the option to override these based on the analyses being performed. A key feature we support for exploratory analyses is similarity-based analyses. Queries such as "more like this" trigger dynamic composition of the query predicates. We support two distinct mechanisms to detect similarity: instance similarity and longitudinal similarity. Given a spatial extent of interest (the reference), instance similarity queries focus on identifying other spatial extents with observed values for each measure of interest that are within "range" of the area of interest. Longitudinal similarity is based on identifying spatial extents that exhibit similarity over much longer time scales. In this case, each spatial extent is represented by a profile vector that encapsulates the average values for each feature monitored at the particular spatial extent. With each spatial extent now represented using a spatial profile vector, we then cluster these vectors in N-dimensional space. We use longitudinal similarity to perform the unsupervised clustering operation to cluster the spatial profile vectors.

D. Paper Contributions

Our methodology is broadly applicable to visualization driven exploratory analyses of spatiotemporally evolving phenomena and includes a mix of algorithmic and systems innovations including:

1) Support for declarative queries and a novel framework that dynamically generates query predicates with the appropriate spatial scopes, and temporal bookends.
2) Support for discretization and binning of feature values observed at a spatial extent. We do so while constraining arbitrary temporal bounds.
3) Support for layering and apportioning of visualization tasks so that the visualizations are amenable to parallelization and rendering using GPUs.
4) Support for interactive visualizations and exploratory analyses over a large number of spatial extents, features, data sources, temporal and spatial scopes.

5) Assimilation of new data, data sources, and features (chemicals) of interest.
6) Preserving locality to minimize data movements while reducing contention in shared clusters.

E. Organization

Section 2 outlines our system architecture and methodology. Section 3 includes a discussion of the empirical benchmarks, performance profiling, and findings. Section 4 covers related work and section 5 describes our conclusions and future work.

II. Methodology

To facilitate visualization driven analysis over voluminous spatiotemporal datasets, our methodology addresses aspects relating to: (1) data wrangling, (2) managing perceptual limits, (3) facilitating GPU accelerated visualizations, (4) enabling support for expressive queries that facilitate rich analyses, (5) data streaming, (6) graphing at diverse temporal and spatial scales, and (7) analyzing adherence to water quality standards.

A. Data Wrangling [RQ-1, RQ-2, and RQ-3]

The data leveraged in our analyses arrives from different sources including the EPA [14], USGS National Hydrology Database [15], and the US Census Bureau [16]. Our data wrangling schemes are designed to ensure (1) harmonization across diverse datasets, (2) reconciling measurements reported in different units, (3) spatiotemporally aligning observations, and (4) minimizing duplicate processing. A related goal is to ensure that visualizations are not impacted by errors or deficiencies in the data.

From the EPA Water Quality Data dataset we ingest the Sites and Physical/Chemical collections. The Sites collection contains records represented by a geoJSON point (coordinate pair), metadata, and a unique identifier. The Physical/Chemical collection contains records represented by measurement name, unit of measurement, time measured, and an identifier which associates the data with a measurement site. At the time of this writing, this collection contains records starting January 1st 1970 - November 2023 within CONUS (continental United States). In this study, the term *measurement* refers to data points or groups of data points that have been measured: chemical compounds, water temperature, weather patterns, etc. From the EPA's EnviroAtlas dataset we ingested the HUC12 watershed boundary collection. This collection contains geoJSON data representing watershed boundaries. We ingested two shapefile collections from the NHD, which has been federated by the USGS: Hydrography and Flowlines. The Hydrography collection contains records (encoded using geoJSON) representing bodies of water such as lakes, reservoirs, rivers, etc. that are represented as multi-polygons. The Flowlines collection contains records (also, encoded in geoJSON) representing water flow lines as MultiLineStrings. When we use the term water bodies, we refer to records from these two collections. To support analyses based on administrative boundaries, we ingested the most recently updated Counties and Tracts shapefile collections from the U.S. Census Bureau. We use the EPA's Water Quality Standards dataset, based on the EPA Clean Water act, for analysis of water quality standard violations.

a) Daily Data Ingestion: We ingest new data from the EPA's Water Quality Physical/Chemical and Sites collections daily. Our auto-ingestion script runs at midnight to ensure that the visualizations assimilate the most recent data. Our data retrievals are targeted and avoid retrievals of data that have been ingested within the system. We communicate with the EPA's servers to retrieve all new data. After performing unit coercion and duplicate/empty values handling the data are staged. To identify if new measurement sites have come online, the collected data is aggregated across Site Id and the resulting set (Set A) is compared to the set of Site Id's already in our database (Set B). We take the difference of these two sets (Set A - Set B) to find all new sites. We ingest the resulting site data and the staged measurement data into our production datasets. Our ingestion script is containerized using Docker and Kubernetes to ensure that it remains online.

b) Associating Measurements with Shapefiles: The association between measurements and shapefiles representing spatial extents (water bodies, watershed boundaries, counties, or tracts) is a foundational component of our data visualization. The spatial component of all our visualizations come from shapefiles and the data being visualized are measurements. Measurements have an implicit association with the station they were taken at. Because our measurement and station data come from a different source than our shapefiles, we need to build associations between the two sources. Specifically, each station needs a reference to each shapefile that it resides within or is proximate to. This involves extensive use of inclusion, exclusion, intersections, and proximity calculations. All distance calculations in aQua are based on spherical coordinates and account for the earth's curvature.

c) Data Cleaning: A majority of the Hydrography and Flowlines shapefiles do not have any associated measurement data. To prune the search space during query evaluations and data retrievals, we defined new collections representing a subset of these shapefiles. These subsets, one for Hydrography and another for Flowlines, contain only those shapefiles with associated measurement data. Henceforth when we reference Hydrography or Flowlines we refer to these new, smaller collections whose shapefiles have associated measurement data.

d) Coercing Measurement Units: In order to improve data availability (the number of measurements taken during a given temporal window for a set of spatial extents), we perform unit coercions when possible. Our unit coercion typically involved converting measurement values to the smallest metric unit used in the dataset with some exceptions. In some corner cases, we converted units to a larger value, for example we converted pg/l to ng/l. To preserve accuracy, we never coerced a unit if that coercion would result in greater than a 3 order of magnitude conversion. We allow users a choice of unit coercion or not; this is possible because we do not alter the original measurement values or unit, rather we added new fields for coerced values and units.

TABLE I
TOP 10 COERCED UNITS

Original Unit	**Converted To**	**Records**
mg/l	ug/l	69,272,262
m	cm	3,745,295
ft	cm	2,958,144
ft3/s	m3/sec	1,604,301
mg/kg	ug/kg	1,333,580
deg F	deg C	916,404
ppm	ppt	739,802
mph	km/hr	697,310
mg/L	ug/l	548,199
cfs	m3/sec	511,192

Unit coercion improves data availability because when we aggregate measurement data for visualization we differentiate between the same pollutant/chemical being measured in different units. This is important because we cannot convert units on the fly during visualization especially when units may be nonstandard or under specified; this is often the case for the EPA Water Quality Physical/Chemical dataset. For example, consider Phosphorus with the following units and associated number of measurements: [mg/l: 122,453, ug/l: 1,594,356, NONE: 532, stdUnits: 1,523, COLSTRAT 3: 3,285]. If we convert mg/l to ug/l we stand to gain 122,453 datapoints every time the user wants to visualize Phosphorus. For obvious reasons we cannot do any conversions for NONE, stdUnits, or COLSTRAT 3 but we do not remove these observations because they could be meaningful to the organizations that collected them. Table 1 displays the top 10 units we coerced along with their respective numbers of coerced datapoints. We coerced 66 units totaling 84,994,949 datapoints.

e) Duplicate/Empty Values Handling: Within the EPA's Physical/Chemical dataset we find that many rows have values that are unusable. There are two cases that require special handling during ingestion. Case-1 is when there is no value for the measurement value column. Case-2 describes situations when there are multiple rows with identical measurement time, Site ID, and measurement name. This is problematic because it suggests that the same compound was measured at the exact same time and at the same place multiple times. We posit that this may occur when organizations are collecting the same data using a variety of collection techniques or sensor types.

In the event of Case-1, we discard the row. Case-2 necessitates more careful handling in order to increase data availability. We attempt to preserve measurements meeting the criteria for Case-2 by coercing units and averaging duplicate values when possible.

B. Managing Perceptual Limits [RQ-1, RQ-2]

Our visualization and data analysis is structured as a web Mercator projection based visual analytics tool that executes within web browsers. As such, interactions with the tool are likely to trigger data movements that may adversely impact interactivity and responsiveness. We have explored two complementary methods to the problem of rendering voluminous spatial data: shape simplification and perceptual limits.

a) Shape Simplification: Complex shapefiles are defined as N-sided polygons represented by tens of thousands of coordinate pairs. In the context of water data these are usually shapefiles defining shorelines of large, irregular water bodies e.g., Lake Michigan, Lake Superior. A single complex shapefile may constitute megabytes of data. Trying to render complex shapefiles in the user's web browser can be problematic because of the computational overheads and accompanying latencies for rendering them at ever increasing precisions.

b) Surface Area Based Rendering: Front-end visualizations occur in the context of a map. This map may be zoomed in and out. As a user zooms out to visualize a much larger aggregate spatial extent, shapefiles below a certain surface area become impossible to discern. Trying to render more than a few thousand shapefiles at once may crash the user's web browser. We analyzed the surface area distribution in the Hydrography collection and found the following statistics for the distribution of the spatial area: Min=0.0 km^2 Max=82,002.288 km^2, Mean=9.031 km^2, and Standard Deviation: 549.182 km^2. Furthermore, we find that 30,021 records (81%) have a surface area less than 1 km^2. This demonstrates a significant left-skew of these data with respect to surface area; see [Fig. 1].

To address this, we define a set of perceptual limits represented by a zoom threshold and minimum visual surface area pair. If the map crosses either one of these thresholds we automatically filter out all shapefiles below the minimum surface area. All surface areas are measured in square kilometers. For example, consider the following $\{$threshold: minimum surface area$\}$ mapping: $\{13 : 0.001\}$, $\{11 : 0.01\}$, $\{9 : 0.1\}$, $\{7 : 1\}$. When the map is zoomed out and crosses zoom level 9, we filter out all shapefiles with a surface area less than 0.1. In other words, we only query and render shapefiles with a surface area ≥ 0.1 km^2. In this way we effectively constrain the search space to the subset of shapefiles which will be visible, and therefore useful, to the user.

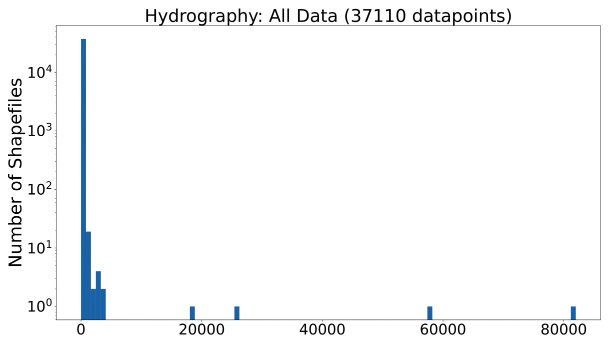

Fig. 1. Surface Area Distribution for Hydrography Collection. The y-axis is plotted on a logarithmic scale and the x-axis is plotted on a linear scale.

C. GPU Accelerated Visualizations [RQ-1, RQ-3]

Our software leverages GPU acceleration via the DeckGL mapping framework. DeckGL is designed to simplify high-performance, WebGL (Web Graphics Library) [17] based visualization of complex datasets. DeckGL's layers accomplish rasterization by emulating 64 bit floating point computations on the GPU. DeckGL exposes a number of useful layer types optimized for handling both point-based and shapefile-based data. Since our map-based visualizations depend primarily on coloring and extruding shapefiles we were able to use these layer types to simplify the parallelization of rendering tasks.

D. Expressive Queries [RQ-2 and RQ-3]

Queries underpin our analyses. We include support for: (1) visual composition of queries, (2) longitudinal similarity, and (3) instance-based similarity. We also support indexing subsets of features to facilitate fast evaluation of queries.

Users can visually compose queries and render outcomes of those queries. The query composition includes support for dynamically refining the spatial scope of the queries (to the viewport), the temporal bounds of interest, and the ranges associated with the feature values based on summary statistics computed during the ingestion phase. Our methodology includes support for indexing a subset of features and generation of compound indices based on features most commonly used during analysis.

Longitudinal similarity identifies spatial extents that are similar to each other over much longer time scales. We leverage an unsupervised learning technique to inform our longitudinal similarity: clustering. Once the temporal bounds of interest are identified, we represent each spatial extent with a feature vector. We leverage longitudinal similarity to cluster water data across a feature vector of the user's choosing. This is useful for identifying areas with unusually high or low values for certain features. For example, a user could choose [Sodium, Chloride, Calcium] as the feature vector and identify spatial extents with outlying averages for these compounds. A user could also use this feature to find spatial extents with spiking averages for lead, arsenic, and other toxic chemicals that can leach into water sources. The clustering operation outputs a list of clusters each of which has a centroid and a list of spatial extents with associated averages for each feature in the vector.

E. Streaming [RQ-1 and RQ-3]

Interactions with aQua often trigger queries that are composed on the fly and evaluated server-side. Our server-side encompasses a distributed set of machines. As queries are evaluated on the server-side, results become available. Rather than wait for results to be available prior to returning them to the client (or even waiting for all results to be available before rendering them) we stream them back to the client. This has two key advantages: responsiveness and the ability to cancel responses (or evaluations) in case the application no longer needs them. Streaming also amortizes the I/O and rendering costs while ensuring that the users experience a responsive tool. This also allows the server-side to scale to support multiple clients and the concurrent query evaluations that this will entail. Crucially, our streaming approach allows us to cancel responses (in cases where the queries trigger a large number of responses) as the user navigates away from the ongoing analysis and launches a new analysis.

F. Graphing at Micro and Macro Scales [RQ-2 and RQ-3]

After loading spatial extents into the map, users can visualize time series line charts of all measurement data associated with the spatial extents. This visualization can be aggregated across multiple selected spatial extents or specific to a single spatial extent. The user interface (UI) is integrated such that users can also easily view comparisons between the selected spatial extents.

The time series analysis feature also displays data availability along the same time series. Users can visually compare observed measurement values and data availability simultaneously. This is important because data availability informs confidence in the averages for a given temporal bucket. If the current temporal window is wide, a single bucket may comprise several months of data. Data availability matters because the average of a bucket with 3,000 data points provides a higher degree of confidence than the average of a bucket with 2 data points.

G. Water Quality Standard Analysis [RQ-1]

This feature allows users to easily visualize which spatial extents do or do not conform to a given water quality standard. Users select a standard (default is EPA 304(a) Recommended Criteria), then choose a pollutant to visualize water quality standards for. We aggregate all measurements for the selected pollutant, grouping by the selected spatial extent. We return 5 metrics associated with water quality standards: Total Exceedences, Rate of Exceedence, Average Exceedence Percent, Mean, and Data Availability. The Total Exceedences represent the number of measurements which exceed the standard. Data Availability is the total number of measurements. The Rate of Exceedence is the Total Exceedences divided by the Data Availability. Average Exceedence Percent is the average percent above the standard for all exceedences. For example, if the standard is 10 and there are 3 exceedences: [12, 13, 14], the average exceedence would be 13. In this example, the Average Exceedence Percent would be 30%. The mean is just the mean value for the pollutant.

III. System Benchmarks

In our implementation, we leverage gRPC to facilitate client/server communication. We use a Flask server and our client is written in JavaScript with React Hooks, ReCharts, and DeckGL to support GPU accelerated map rendering. We leverage a distributed data store based on MongoDB and MapReduce via aggregation pipelines for data processing. Our systems benchmarks profile ingestion times, server-side evaluation of queries, interactivity assessments and shape rendering time using the Google Lighthouse tool. We executed

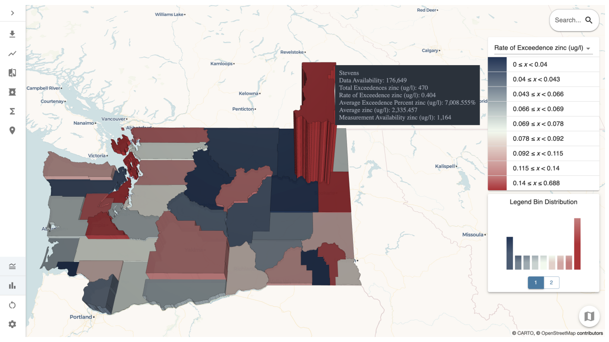
Stevens
Data Availability: 176,649
Total Exceedences zinc (ug/l): 470
Rate of Exceedence zinc (ug/l): 0.404
Average Exceedence Percent zinc (ug/l): 7,008.555%
Average zinc (ug/l): 2,335.457
Measurement Availability zinc (ug/l): 1,164

Fig. 2. Analyzing water quality violations for Zinc in Washington State. Counties are colored based on *Rate of Exceedence* and extruded based on *Average Exceedence Percent*. We can easily see that Stevens County has an extreme *Average Exceedence Percent* (7000%) and a high *Rate of Exeedence* (40%) This means that 40% of measurements for Zinc are in excess of the water quality standard, and on average these exceedences were 7000% above the standard.

all benchmarks on a set of 50 Hewlett-Packard DL160-G6 machines equipped with a 6-core 2.4 GHz Intel Xeon CPU E5-2620 v3 processor and 64 GB RAM. Each machine currently runs Alma Linux with the 9.2 kernel.

a) Ingestion Benchmarks: We profiled the times involved in targeted retrievals of new data from the EPA servers, while accounting for new monitoring stations (data sources) and types of chemicals being tracked. We tracked ingestion times when the time gaps from the most recent ingestion was a day, a week, and a month. The observed ingestion times were 562 ms (for a day), 842 ms (for a week), and 1214 ms (for a month). The ingestion times were dominated by communication latencies.

b) Query Latencies: We profiled query evaluation latencies for several of our queries. We benchmarked each query made to the server-side 30 times, and report the mean and standard deviations for each query type. Several of our queries are generalized to handle different versions of the same query. For example, the measurements query can find measurements associated with a set of spatial extents, measurements to build a feature vector, or measurements with water quality standards. Both mean and standard deviation are reported in milliseconds. Because our queries can be performed across such a wide variety of spatial extents we provide two benchmarks for many of our queries at different spatial extents: one for the state of Colorado and one for the continental United Sates (CONUS). The temporal range for all of the reported benchmarks was 53 years. These benchmarks can be seen in Table II.

c) Google Lighthouse Benchmarks: We used Google Chrome's Lighthouse tool [19] to benchmark client interactivity and to compare rendering times using a GPU accelerated map framework (DeckGL) with a standard CPU based map framework (leaflet). We report average blocking time for various rendering tasks in Table III and average blocking time for GPU vs CPU rendering in Table IV and Fig 3. All blocking times are reported in milliseconds. Our benchmarks illustrated in Table III demonstrate that aQua is interactive despite vast spatiotemporal scope of the data that underpins the visualization and analysis and our benchmarks illustrated in Table IV demonstrate that GPU acceleration becomes increasingly crucial with broader spatial extents.

TABLE II
QUERY EVALUATION LATENCIES IN MILLISECONDS

Query	Mean	Std Dev	Spatial Scope
Shapes	996.71	688.27	813 Water Bd.
MoreLikeThis	914.63	2,604.58	CONUS
Chart	560.65	305.24	813 Water Bd.
Chart	947.14	461.88	Colorado
SiteOrg	619.49	623.60	813 Water Bd.
SiteOrg	2,853.65	749.58	Colorado
Comparison	749.73	235.31	813 Water Bd.
Comparison	2,187.26	569.72	Colorado
Longitudinal Similarity	73,785.14	9,820.79	CONUS
Measurements	824.42	327.55	813 Water Bd.
Measurements	12,834.84	1,524.74	Colorado

TABLE III
GOOGLE LIGHTHOUSE BENCHMARKS

Task	Blocking Time	Spatial Data Complexity
Render Shapes	230	873 WB
Overview Feature	0	873 WB
Load Measurements	53.33	873 WB, 79,955 M
Time Series Chart	423.33	873 WB, 79,955 M
Shape Comparisons	236.67	873 WB, 79,955 M
More Like This	33.33	15,314 M
Water Quality Standards	0	151,557 M

Blocking times (milliseconds) reported by Lighthouse. WB: Water Bodies, M: Measurements

TABLE IV
CPU VS GPU BENCHMARKS

# Shapes Rendered	CPU Blocking Time	GPU Blocking Time
30	110	0
300	170	90
3,000	3,500	970

d) Server Benchmarks: We briefly describe these queries: (1) the shapes query is responsible for retrieving all spatial extents within the map view port bounds and calculate data availability per spatial extent, (2) the "More Like This" query identifies spatial extents that match a similarity index computed based on user-specified preferences, (3) the chart query is responsible for constructing chartable data by aggregating measurement values, bucketing, and averaging. (4) the SiteOrg query is responsible for identifying organizations that perform data collection activities at a set of spatial extents, (5) the comparison query is responsible for averaging and normalizing measurement values across a collection of shapefiles, (6) the longitudinal similarity query performs the unsupervised k-means clustering [18] operation based on the user-specified set of features, and (7) the measurements query

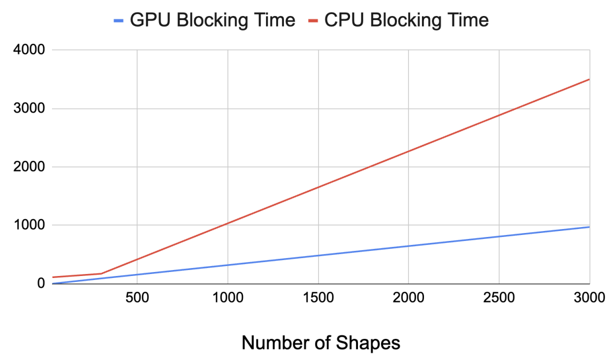

Fig. 3. Due to the spatial density of water bodies in many regions throughout CONUS, it is common to render thousands of water bodies on the map at once. Interactivity in these spatial regions necessitates the visualization speed-ups available using a GPU accelerated map framework.

can be parametrized to retrieve measurements associated with spatial extents.

Our server-side queries (Table II) indicate that data exploration at the water body level usually returns results with sub-second latencies to ensure interactivity. Collate this with the client benchmarks (Table III) and we find that data rendering time is under half a second. Data exploration at the state level can produce longer query times because, depending on the state, the server may be performing computations across hundreds of millions of data points. The longitudinal similarity query takes as long as it does because it is considering every data point associated with every body of water in the entire CONUS; further, the k-means clustering underpinning this query is also performed in the critical path.

IV. Related Work

Rose and Hildebrand [20] leverage WebGL to facilitate GPU Accelerated rendering of 3D molecular structures. By calling WebGL's API from the browser the authors are able to send complex rendering tasks directly to the GPU. Perrot et al. [21] describe browser-based visualizations optimized for GPU acceleration using WebGL with a focus on generating heatmaps at scale. Xie et al. [22] discuss the importance of GPU acceleration in the context of large-scale data visualization. Their research does not visualize data in the context of a web browsers and thus does not involve WebGL, though it does necessitate interactivity. The authors chose to use a GPU accelerated approach to maintain interactivity which is often lost in distributed computing settings. Li et al. [23] suggest that visualization can be defined as a function mapping data to visual primitives. Their framework ensures the availability of visualizations using a threshold which defines the minimum size of visual primitives.

We chose to use a WebGL based mapping framework for similar reasons that [20] and [21] chose to integrate WebGL into their research. We leveraged a WebGL-based framework (DeckGL) for GPU acceleration while harnessing mapping layer types we need for our visualizations. Crucially, the use of DeckGL frees up the CPU to maintain user interaction while rendering is taking place. While [23] defined a threshold for the minimum size of visual primitives we defined a threshold for the minimum surface area of water bodies to ensure the user's browser wouldn't crash and that we were not rendering information that is not visible to the user.

Mayorga and Gleicher [24] describe their approach to visualizing voluminous spatial data in scatter plots by defining perceptual limits which define the maximum data density visible in a given area of the screen. Healey and Sawant [25] discuss their work on perceptual level-of-detail limits for visualizing data. To avoid performing wasteful computations the authors sought to define a process for determining when there are not enough pixels on the screen to render a visual element. Liu et al. [26] present methods for maintaining interactivity in systems designed to visualize big data. Their methods follow the principle that perceptual limits determine the scalability of visual information. Their experiments were conducted in the context of a web browser-based application which used WebGL to facilitate GPU accelerated rendering. Harrison et al. [27] attempt to determine the effectiveness of perceptual limits in defining visualization design using Weber's Law to quantify the perceived change in a given stimulus and to compare perceptual precision.

Our work is complimentary to the work described in [24] and [25] because we also implemented perceptual limits in order to deal with problems relating to interactivity in the context of vast spatial data rendering. Our work differs in the exact implementation of perceptual limits. Where [24] defined maximum data density and [25] identified minimum numbers of pixels we mapped zoom thresholds to minimum water body surface areas. This implementation of perceptual limits allows us to cut out computation relating to water bodies which would be imperceptible to the user. Further, the impact on interactivity in [27] validates our use of perceptual limits.

Stream processing frameworks such as Spark [28], Storm [29], and Samza [30] have been used to support efficient streaming in several applications. Our choice of using a Flask server was driven primarily by its lightweight properties (lower memory footprints and reduced computational overheads) that though not quite as full-featured, meets all our functional and, more importantly, performance requirements.

V. Conclusions & Future Work

In this study we described our methodology to facilitate visualization-driven exploratory analyses over voluminous spatiotemporal data collections.

RQ-1: Dynamically pruning the rendering complexity of the visual elements based on shape simplifications and perceptual limits during drill-down and roll-up operations reduces computational requirements while preserving interactivity. Prefetching data to account for boundary conditions around the viewport allows us to ensure responsiveness during panning operations. Streaming and rendering visual elements allows incremental rendering, amortizes rendering overheads, and preserves responsiveness.

RQ-2: Our preprocessing tasks are expressed such that they have data locality reducing network I/O and contention. Accounting for perceptual limits allows us to avoid streaming and rendering visual elements. More importantly, this reduces the amount of I/O that needs to be performed. To support data retrievals at scale, we preferentially index a subset of features based on their usage and include support for compound indices for features that are often used in tandem with each other. This allows faster query evaluations while conserving memory.

RQ-3: Declarative queries and dynamic query generation simplify the complexity of analyses tasks. Our similarity analysis allows users to specify features of interest and leverage unsupervised learning to cluster spatial extents in a multidimensional space. Users also have the ability to configure and override several aspects. Finally, our discretization and binning capabilities allow users to analyze feature value changes compactly and at scale. Leveraging GPU accelerated visualizations on the client-side allows our framework to be responsive and scale with a large number of users.

As part of future work, we propose to explore imputation schemes based on spatiotemporal variation in the pollutants. Our current work lays the groundwork for identifying data sources that are malfunctioning based on the rates, times, and gaps in data availability. We propose to fit lightweight timeseries models over data at different water bodies. Because these timeseries models can capture trends and seasonality in the data, they can be used to both forecast expected pollutant values and to impute values where gaps exist.

VI. Acknowledgement

This research was supported by the National Science Foundation (OAC-1931363, CNS-2312319), and an NSF/NIFA Artificial Intelligence (AI) Institutes AI-CLIMATE Award [2023-03616].

References

[1] Nativi, S., Mazzetti, P., Santoro, M., Papeschi, F., Craglia, M. and Ochiai, O., 2015. *Big data challenges in building the global earth observation system of systems.* Environmental Modelling & Software, 68, pp.1-26.

[2] Mitra S, Khandelwal P, Pallickara S, Pallickara SL. *Stash: Fast hierarchical aggregation queries for effective visual spatiotemporal explorations.* In2019 IEEE International Conference on Cluster Computing (CLUSTER) 2019 Sep 23 (pp. 1-11). IEEE.

[3] Koontz, J., Malensek, M. and Pallickara, S., 2014, December. *Geolens: Enabling interactive visual analytics over large-scale, multidimensional geospatial datasets.* In 2014 IEEE/ACM International Symposium on Big Data Computing (pp. 35-44). IEEE.

[4] Rammer, D., Lee Pallickara, S. and Pallickara, S., 2019, December. *Atlas: A distributed file system for spatiotemporal data.* In Proceedings of the 12th IEEE/ACM International Conference on Utility and Cloud Computing (pp. 11-20).

[5] Aji, A., Wang, F., Vo, H., Lee, R., Liu, Q., Zhang, X. and Saltz, J., 2013. *Hadoop gis: a high performance spatial data warehousing system over mapreduce.* Proceedings of the VLDB Endowment, 6(11), pp.1009-1020.

[6] Eldawy, A. and Mokbel, M.F., 2015, April. *Spatialhadoop: A mapreduce framework for spatial data.* In 2015 IEEE 31st international conference on Data Engineering (pp. 1352-1363). IEEE.

[7] Yu, J., Wu, J. and Sarwat, M., 2015, November. *Geospark: A cluster computing framework for processing large-scale spatial data.* In Proceedings of the 23rd SIGSPATIAL international conference on advances in geographic information systems (pp. 1-4).

[8] Hanrahan, P., 2012, May. *Analytic database technologies for a new kind of user: the data enthusiast.* In Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data (pp. 577-578).

[9] Heer, J. and Shneiderman, B., 2012. *Interactive dynamics for visual analysis.* Communications of the ACM, 55(4), pp.45-54.

[10] Bruhwiler, K. and Pallickara, S., 2019, December. *Aperture: Fast visualizations over spatiotemporal datasets.* In Proceedings of the 12th IEEE/ACM International Conference on Utility and Cloud Computing (pp. 31-40).

[11] Van Wijk, J.J., 2005, October. *The value of visualization.* In VIS 05. IEEE Visualization, 2005. (pp. 79-86). IEEE.

[12] Powers of 10: Time Scales in User Experience, *Powers of 10: Time Scales in User Experience* [online]. Available: https://www.nngroup.com/articles/powers-of-10-time-scales-in-ux/

[13] Project Sustain, *aQua*, Colorado State University [online]. Available: https://urban-sustain.org/aqua-client/

[14] United States Environmental Protection Agency, *Water Quality Physical/Chemical*, US EPA [online]. Available: https://www.epa.gov/waterdata/water-quality-data-download

[15] United States Geological Survey, *National Hydrology Database*, USGS [online]. Available: https://www.usgs.gov/national-hydrography/access-national-hydrography-products

[16] United States Census Bureau, *United States Counties*, US Census [online]. Available: https://www.census.gov/geographies/mapping-files/time-series/geo/tiger-line-file.html

[17] Parisi, T, *WebGL: up and running*, O'Reilly Media, Inc.

[18] Sinaga, K.P. and Yang, M.S., 2020. *Unsupervised K-means clustering algorithm.* IEEE access, 8, pp.80716-80727.

[19] Heričko, T., Šumak, B., & Brdnik, S., *Towards Representative Web Performance Measurements with Google Lighthouse*, 2021, September, In Proceedings of the 2021 7th Student Computer Science Research Conference (p. 39).

[20] Alexander S. Rose , Peter W. Hildebrand, *NGL Viewer: a web application for molecular visualization*, Nucleic Acids Research, Volume 43, Issue W1, 1 July 2015, Pages W576–W579.

[21] A. Perrot, R. Bourqui, N. Hanusse, F. Lalanne and D. Auber, *Large interactive visualization of density functions on big data infrastructure*, 2015 IEEE 5th Symposium on Large Data Analysis and Visualization (LDAV), Chicago, IL, USA, 2015, pp. 99-106, doi: 10.1109/LDAV.2015.7348077.

[22] J. Xie, F. Sauer and K. -L. Ma, *Fast uncertainty-driven large-scale volume feature extraction on desktop PCs*, 2015 IEEE 5th Symposium on Large Data Analysis and Visualization (LDAV), Chicago, IL, USA, 2015, pp. 17-24, doi: 10.1109/LDAV.2015.7348067.

[23] X. Li, A. Kuroda, H. Matsuzaki and N. Nakajima, *Advanced aggregate computation for large data visualization*, 2015 IEEE 5th Symposium on Large Data Analysis and Visualization (LDAV), Chicago, IL, USA, 2015, pp. 137-138, doi: 10.1109/LDAV.2015.7348086.

[24] A. Mayorga and M. Gleicher, *Splatterplots: Overcoming Overdraw in Scatter Plots*, 2013 EEE Transactions on Visualization and Computer Graphics, vol. 19, no. 9, pp. 1526-1538, Sept. 2013, doi: 10.1109/TVCG.2013.65.

[25] Healey, Christopher G. and Sawant, Amit P., *On the Limits of Resolution and Visual Angle in Visualization*, 2012 Association for Computing Machinery, New York, NY, USA, vol. 9, no. 4, 1544-3558, doi: 10.1145/2355598.2355603.

[26] Liu, Z., Jiang, B. and Heer, J, *imMens: Real-time Visual Querying of Big Data*, Computer Graphics Forum, 32: 421-430.

[27] L. Harrison, F. Yang, S. Franconeri and R. Chang, *Ranking Visualizations of Correlation Using Weber's Law*, in IEEE Transactions on Visualization and Computer Graphics, vol. 20, no. 12, pp. 1943-1952, 31 Dec. 2014, doi: 10.1109/TVCG.2014.2346979.

[28] Zaharia, M., Das, T., Li, H., Hunter, T., Shenker, S., & Stoica, I, *Discretized streams: Fault-tolerant streaming computation at scale*, 2013, November, in Proceedings of the twenty-fourth ACM symposium on operating systems principles (pp. 423-438).

[29] Kulkarni, S., Bhagat, N., Fu, M., Kedigehalli, V., Kellogg, C., Mittal, S., ... & Taneja, S, *Twitter heron: Stream processing at scale*, 2015, May, in Proceedings of the 2015 ACM SIGMOD international conference on Management of data (pp. 239-250).

[30] Noghabi, S. A., Paramasivam, K., Pan, Y., Ramesh, N., Bringhurst, J., Gupta, I., &; Campbell, R. H., *Samza: stateful scalable stream processing at LinkedIn*, 2017, Proceedings of the VLDB Endowment, 10(12), 1634-1645.