Published in final edited form as:

*Trans GIS*. 2017 June ; 21(3): 546–559. doi:10.1111/tgis.12286.

# Terra Populus' Architecture for Integrated Big Geospatial Services

**David Haynes**,
Minnesota Population Center University of Minnesota, Minneapolis, MN

**Steven Manson**,
Department of Geography, Environment, and Society University of Minnesota, Minneapolis, MN

**Eric Shook**
Department of Geography, Environment, and Society University of Minnesota, Minneapolis, MN

David Haynes: dahaynes@umn.edu; Steven Manson: manson@umn.edu; Eric Shook: eshook@umn.edu

## Abstract

Big geospatial data is an emerging sub-area of geographic information science, big data, and cyberinfrastructure. Big geospatial data poses two unique challenges to these and other cognate disciplines. First, raster and vector data structures and analyses have developed on largely separate paths for the last twenty years and this creates an impediment to researchers utilizing big data platforms that do not promote the integration for these classes. Second, big spatial data repositories have yet to be integrated with big data computation platforms in ways that allow researchers to spatio-temporally analyze big geospatial datasets. IPUMS-Terra, a National Science Foundation cyberInfrastructure project, begins to address these challenges. IPUMS-Terra is a spatial data infrastructure project that provides a unified framework for accessing, analyzing, and transforming big heterogeneous spatio-temporal data, and is part of the IPUMS (Integrated Public Use Microdata Series) data infrastructure. It supports big geospatial data analysis and provides integrated big geospatial services to its users. As IPUMS-Terra's data volume grows, we seek to integrate geospatial platforms that will scale geospatial analyses and address current bottlenecks within our system. However, our work shows that there are still unresolved challenges for big geospatial analysis. The most pertinent is that there is a lack of a unified framework for conducting scalable integrated vector and raster data analysis. We conducted a comparative analysis between PostgreSQL with PostGIS and SciDB and concluded that SciDB is the superior platform for scalable raster zonal analyses.

## 1. Introduction

Many challenges humanity faces today are interconnected, complex and often involve interactions among human and environmental systems. Big data is viewed as a new tool that should be embraced by the larger scientific community in an effort to solve complex global problems (Hampton et al., 2013; Mayer-Schönberger & Cukier, 2013; Murdoch & Detsky, 2013). In particular, we face a growing array of complex or "wicked" problems, so-called

Correspondence to: David Haynes, dahaynes@umn.edu.

because they are dynamically complex and ill-structured (Batie, 2008; Brown, Harris, & Russell, 2010). These problems are typically rooted in a social context and involve multiple stakeholders with differing perspectives; therefore, optimal solutions are often elusive. Additionally, many of these challenges are due to economic and population changes that have occurred in the last five decades. In this time, the world's population has more than doubled, resulting in alarming environmental degradation and global climate change (O'Neill, MacKellar, & Lutz, 2005; World Bank, 2012). Population growth and human behavior playing out within a larger social and technological context lie at the heart of a myriad of environmental challenges facing the planet, as well as necessarily being part of their solutions.

Heterogeneous big data repositories that leverage high performance computation frameworks are necessary for advancing research discoveries and solving complex human-environment challenges. These repositories help overcome the challenge posed by data silos, the existence of which hamper our understanding of these complex problems. Specifically, understanding the interactions among population and environmental systems is hampered by the dearth of integrated data. In order for scientists, students, and policy makers to tackle many challenges, they need fast and easy access to big data. However, there are critical data and computational infrastructure gaps in using big data to address these challenges. Geospatial researchers need frameworks that utilize and understand spatial data structures, support spatial analysis, and utilize spatial data visualizing techniques. These frameworks will allow us to understand spatio-temporal relationships between the world's population and environment, and provide unprecedented opportunities to investigate the agents of change, assess their implications for human society and the environment, and develop policies to meet future challenges.

## 2. Background

Existing data repositories and cyberinfrastructures have made significant progress in advancing GIScience research, yet gaps remain. One basic challenge is that the volume of data is growing. These datasets are siloed, trapped in domain-specific infrastructures, formats, or storage locations. With the advent of large-scale spatial data infrastructure, users have greater access to data and analysis capability, but a challenge for many users is moving data to, from, or between these infrastructures. A final challenge facing researchers and others in using big spatial data is the need for analytical capabilities that work across data structures, particularly vector and raster formats.

The primary challenge with big geospatial data is simply the sheer amount of data available. Every day Facebook generates over half a petabyte of data (Vagata & Wilfong, 2014). Digital Globe collects approximately two petabytes of satellite imagery each year (Babcock, 2013), and a single self-driving car is estimated to generate two petabytes of data a year (van Rijmenam, 2016). Research communities also play a role in producing big data. The Intergovernmental Panel on Climate Change (IPCC) recent Coupled Modeled Intercomparison Project (CMIP5), for example, is approximately two and half petabytes (Taylor, Stouffer, & Meehl, 2012). NASA's Shuttle Radar and Topographic Mission (SRTM) 90 meter digital elevation of the world is half a terrabyte (Jarvis, Reuter, Nelson, Guevara, &

others, 2008). These are just a few examples of the kinds of data that are becoming necessary to address a range of research questions.

Another challenge is that many of these datasets originate from different academic and commercial communities, resulting in big data that is siloed across many domains. In order to address wicked problems, however, we need integrated data. Population data integrated with data on the environment gives us the opportunity to describe the unfolding transformation of human and ecological systems. Indeed, data on the human population are crucial for understanding the impact of society on biological and climate systems. Equally important are climate and high resolution landcover datasets that provide essential information for understanding the effect of the environment upon the human population. For example, changes in global temperature are altering the endemic regions of disease and affecting new populations and the data needed to understand these relationships comes from multiple human and environmental sources and exists and various spatial scales. Access to big integrated data is essential for analyzing today's complex problems.

A significant challenge for big spatial data users is moving massive datasets to new infrastructures. Spatial data infrastructures (SDIs) such as the United States Geological Survey (USGS) National Map represent major efforts to build easy-to-use web-based interfaces for big data repositories (United States Geological Survey, 2016). Similar to other SDIs, the National Map focuses on disseminating data and provides few data analysis tools. To analyze and synthesize spatial data, users of SDI rely on additional tools such as desktop GIS. However, as the size of spatial data increases, these desktop-based tools are becoming inadequate for handling the computational demands. New web-based portals such as the CyberGIS Gateway, Geospatial Building Blocks (GABBs), and Google Earth Engine serve as advanced cyberinfrastructure providing geospatial data tools and services (CyberGIS Gateway, 2016; Geospatial Data Analysis Building Blocks, 2016; Google Earth Engine, 2016). Unlike SDIs, many of these portals focus on computational and analytical capabilities and often rely on users to provide a much of spatial data that are analyzed. There are significant technical and logistical challenges involved in transferring and synthesizing a range of geospatial datasets that may be explicitly spatial, such as raster and vector data, or implicitly spatial, such as tabular data with street addresses. A single infrastructure that combines the features of SDI and CyberGIS would offer an elegant end-to-end solution for users working with massive spatial data and circumvent some of the current limitations of desktop-based GIS software.

Processing big geospatial data, whether raster or vector, remains a challenge for geospatial scientists. For decades geospatial scientists have continually leveraged the latest technologies to solve the most pressing geospatial problems (Dobson, 1983). There have been significant advances in the use of computing technologies to help tackle these challenges under various evolving terms including distributed, grid, cloud, and data-intensive computing. Distributed and grid computing in particular offer significant computational resources for geospatial problem solving (C. Yang & Raskin, 2009; Zhang & Tsou, 2009). Hofer (2015) provides a systematic review of online geoprocessing studies and platforms. Cloud computing platforms such as Microsoft Azure and Google Earth Engine provide a compelling means to advance geospatial sciences (C. Yang et al., 2011). Yue, et al.

(2013) compares two popular cloud platforms, Microsoft Azure and Google App Engine, using a suite of geoprocessing methods and demonstrates the growing utility of these systems while also showing they fall short in some ways. Similarly, cyberinfrastructure has provided a new paradigm for advancing geospatial sciences including spatial cyberinfrastructure (Wright & Wang, 2011), geospatial cyberinfrastructure (C. P. Yang & Raskin, 2010), and (cyberGIS Wang, 2010).

Another challenge for spatial big computing is the need for analytical capabilities that work across data structures, particularly vector and raster formats. Spatial data has always been large and GIScience researchers have long focused on developing novel ways of partitioning data that always seem to be a bit larger and more complex than existing computing systems can handle (Güting, 1994; Ding & Densham, 1996; Papadias, Kalnis, Zhang, & Tao, 2001; Zhou, Abel, & Truffet, 1998; Armstrong, 2000). What is being unveiled in the era of big geographic data is that methods for analyzing vector and raster data have evolved separately for the last 20 years, and accordingly, we now have a number of specialized Big Geo Data platforms, such as SpatialHadoop, GeoMesa, GeoTrellis, Greenplum DB, Rasdaman, and SciDB. However, each focuses, for the most part, on either vector or raster data formats and there are no integrated solutions.

Taken together, these challenges present a deep need for high performance spatial computation systems that offer integrated management, querying, and storage of two equally important spatial data formats. Table 1 describes a number of open-source platforms currently available for analyzing big spatial data. Eldawy and Mokbel (2015) and Olasz et al. (2016) provide extensive reviews of big geospatial data platforms. They conclude that big data faces the unfortunate situation of utilizing a collection of systems that each prioritizes a single spatial data type at the cost of working with, or integrating, multiple data types. The focus on a single data structure within a computation platform allows developers to maximize the analytical capacity at the cost of forcing researchers to develop their own techniques for moving data back forth from various data structures. For geospatial researcher the lack of an integrated system imposes a tremendous burden on and inhibits the development of innovative scalable geospatial analyses.

## 2.1 IPUMS-Terra

IPUMS-Terra is a National Science Foundation project that hosts terabytes of socio-economic and environmental data. IPUMS-Terra is a next-generation spatial data repository with two main foci: data integration and data access. Data integration is achieved by providing a unified framework for accessing heterogeneous data: microdata, vector, and raster datasets (Table 2). Data access is facilitated by guided web interfaces that support standardized scientific workflows. The guided interface allows users the ability to access, apply, and integrate new datasets into their research.

IPUMS-Terra is amongst the largest sources of curated global human-environment data, incorporating areal data for geographic units, individual-level microdata, and raster data. It draws on two datasets from the Minnesota Population Center, the National Historical Geographic Information System (NHGIS) and Integrated Public Use Microdata Series projects (IPUMS). NHGIS has the largest database of United States Census aggregate data, with over 265 billion data points, while IPUMS is the largest individual-level population database in the world, with over half a billion person records and a total data volume size of two terabytes. The raster data collection currently provides over 4,000 global rasters, approximately half a terabyte, on agriculture, climate, and land cover.

IPUMS-Terra has three main applications for researchers. The main application is the extract builder (data.terrapop.org), which has a guided web interface that allows users to create customized extracts from our existing data repository. The extract builder allows users to generate extracts that match their desired format. Raster extracts result in compressed (zip) files with georeferenced tiff files. Area-level or vector extracts contain ESRI shapefiles and comma-delimited text files (csv) that can be readily joined together and utilized by GIS software. Microdata extracts are csv files that can be easily loaded into statistical programs. The second application is TerraScope, (data.terrapop.org/terrascope), which is a data exploration platform for visualizing the IPUMS-Terra data collection. TerraScope provides users the opportunity to access, visualize, and analyze thousands of spatio-temporal datasets. Users are able to view the availability of variables across time and space, create choropleth maps of area-level data, and visualize raster datasets. The final application is TerraClip, (data.terrapop.org/terraclip), which allows users to download raster datasets within the collection. Taken together, all three address gaps in current cyberinfrastructure for big data of human-environmental systems.

## 2.2 Integrated Big Geo Services

What differentiates IPUMS-Terra from traditional data repositories or geospatial portals is we present an integrated system that connects large geospatial data to a spatial computation platform. Geospatial portals typically host a number of pre-calculated datasets. Through IPUMS-Terra, however, users make requests for tailored extracts, yielding a compilation of datasets customized by the system and delivered to the end user. Conceptually this is similar to a geoprocessing service, in which the user submits a request and is returned a spatial object.

IPUMS-Terra's architecture (Figure 1) is unique and its purpose is multifaceted. The system must be able to quickly retrieve, analyze, and transform any of its three datasets and return those results to any of its web applications. The user interacts with web our applications (e.g., Extract Builder, TerraScope, TerraClip) to generate user-defined data requests. Those requests are submitted by the web applications to the web layer, where they are translated into a series of database transactional calls, which are handled by the Extract Assembler. The Extract Assembler interacts directly with PostgreSQL and assembles the user-defined request, which is typically a series of datasets and variables. Once the extract is assembled the user is alerted via email notification.

The spatial information is utilized for any data transformation in IPUMS-Terra. We illustrate this through one of our guided workflows, in which raster data is summarized and attached to geographic boundaries for an area level extract. In this example, a researcher requests aggregate population data for the United States, 2010 census. Each geographic feature will then have the percent of the state's area used for agriculture attached to each record. The system uses the land cover dataset and state boundaries to calculate the percentage of each state's area used for agriculture. These values are then attached to the area-level records using the state codes.

The volume and diversity of the IPUMS-Terra data collection dictates that the transformations performed to integrate data must be conducted on-the-fly as they are requested by users. Currently the project has over 780 spatio-temporal boundary datasets with associated population data and over 4,000 different rasters. If we were to pre-generate the vector to raster transformations, we would have to maintain over 3 million rasters. Instead, the system utilizes custom written functions stored within the PostgreSQL database to generate them as users make requests.

IPUMS-Terra's web applications provide access to Web Processing Services (WPS) without requiring the user to have pre-existing knowledge for conducting spatial analysis. The services provided through IPUMS-Terra are specialized WPS designed to handle big spatial data. The IPUMS-Terra system does not use a geospatial data-sharing application (e.g. Geoserver, MapServer) for delivering these big datasets. Because the primary purpose of these applications are web mapping and they have limited tool sets for delivering customized big spatial datasets (Jaeger et al., 2005). These platforms are limited as they require the data to be within their own computation platform. Instead, IPUMS-Terra uses custom written database functions to conduct the analysis where the data are stored. This approach is superior for delivering big spatial data to a broad audience.

### 2.3 IPUMS-Terra's Need for High Performance Spatial Architecture

IPUMS-Terra's initial architecture design is simple in that it utilizes PostgreSQL for all of its data structures. Our reliance on PostgreSQL has merit as it is still the only open-source off-the-shelf system capable of working with both raster and vector data types. PostgreSQL's open-source community makes it the jack of all trades and is therefore easily adaptable to new challenges. Many of the big geo architectures listed in Table 1 were in their infancy in 2012, when this project began, making PostgreSQL the logical choice. However, as IPUMS-Terra moved out of beta testing it became apparent that the architecture would not support an expanding user-base, increasing data requests, or growing raster spatial resolutions. In particular, raster summarizations are one of the largest bottlenecks of the current system and, per above, a general problem for big spatial data. This paper compares two platforms for raster summarizations and develops findings that are broadly applicable to others users and systems that leverage both raster and vector data for large analytical procedures.

Raster summarizations are problematic in IPUMS-Terra because users can submit any combination of vector and raster datasets. For instance, User A requests an extract identifying the percent of urban areas in all counties of the United States from the MODIS Aqua/Terra satellite imagery. User B requests the percent area of deciduous forest for all states in the United States. In the current system these requests would utilize the same stored procedure for analysis, but would result in widely different performance times. The variation in query performance times is due to vector irregularity and the tile size of the raster dataset.

### 2.4 High Performance Geocomputation Environments

Table 1, provides a summary of the current open-source platforms available for conducting high performance geospatial analysis. The current IPUMS-Terra architecture led us to focus our research on two geocomputation environments that operate on different data models. PostgreSQL with PostGIS, operates on the entity model is ideally suited for vector datasets, whereas SciDB's array model should benefit raster analysis.

PostgreSQL with the PostGIS extension is the only robust open-source system built for server-side operations that conducts spatial analysis of both raster and vector data types (as distinct from more desk-top oriented systems such as QGIS and GRASS, although this distinction is blurring over time). PostgreSQL does not natively support parallel queries for analyzing many kinds of big spatial data, and parallelization (essentially breaking a big problem into many small ones handled by separate computing cores) is the primary way in which big data is handled by most computation platforms. In our previous work we extended upon the existing PostgreSQL platform (Haynes, Ray, Manson, & Soni, 2015; Ray, Simion, Brown, & Johnson, 2014). Haynes et al. (2015), discussed a prototype shared-nothing parallel spatial database, called SpatialStado. SpatialStado acts as a coordinating node allowing it to leverage multiple PostgreSQL instances but currently has not implemented the raster data type.

Relational Database Systems (RDBS) like PostgreSQL, which operate on the entity model, are not well designed to handle raster datasets. Stonebraker (2011) describes how RDBS, which operate on a table platform must simulate arrays to handle raster data types, and this simulation comes at a cost. In our experience, this is a major reason why queries that involve high resolution raster datasets may fail within PostGIS.

SciDB is an open-source multi-dimensional array database and parallel computation engine, designed to work with array-structured datatypes such as astronomy data and genomics, and is now being adopted to geospatial datasets. Array databases such as SciDB and Rasdaman are well suited for analyzing large array datasets (Pavlo et al., 2009; Planthaber, Stonebraker, & Frew, 2012). However, there has been no existing literature that we are aware of that test either of these platforms performance in the primary forms of raster spatial analyses: local, focal, and zonal.

Scaling zonal analysis in particular to handle big spatial data is challenging. Unlike many geospatial methods that operate exclusively on raster or vector data, zonal statistics operate across both raster and vector data. As a result, these methods must integrate calculations across massive raster data as well as irregular vector data. Zones are spatially irregular, but the calculations applied to each zone are identical. Such problems are classified as loosely-synchronous by Ding and Densham (1996) because the spatial domain is irregular, but the algorithms (calculations) are regular. This classification of problems, including zonal statistics, can be parallelized using non-overlapping equal area or adaptive spatial partitioning.

## 3. Methods

We compared SciDB and PostgreSQL with PostGIS in terms of how they performed with zonal overlays between vector and raster datasets. This operation is quite common to any geospatial process that involves combining different data types and is essential to the existing bottleneck in IPUMS-Terra. We installed PostgreSQL 9.6 with PostGIS 2.3.0 and SciDB 16.9 onto a single machine with 8 gigabytes of random access memory (RAM) and four processing cores. The operations that we tested in this analysis are the spatial join and array join features of both platforms. As both platforms operate on different data models, we will evaluate which data model is optimal for raster summarizations. Since SciDB does not natively support vector datasets, we utilized secondary software to transform the vector dataset into an array that can be read by SciDB. This array has the same spatial resolution as the raster data and acts as a mask, which is analogous to the vector/raster overlay process in PostgreSQL.

All of the queries written for PostgreSQL and SciDB utilized aggregate functions. Aggregate functions are a class of functions in databases that operate over user-defined sets of data. For example, the function “average” or “sum” in Structure Query Language (SQL) is an aggregate function that can be utilized in a SQL query with a user-defined “Group By” clause. Aggregate functions are important because when written correctly they improve query performance by decomposing mathematical formulas into elements that can be operated on sequentially (thereby reducing memory requirements) or in parallel (reducing execution time). The aggregate function ST_SummaryStatsAgg only recently became available in PostGIS 2.2.0 (Ramsey, P., 2015).

### 3.1 Platform Specifications

PostgreSQL with PostGIS supports both vector and raster dataytpes, both vector datasets and raster datasets are loaded into the database. The vector datasets utilize the Generalized index Search Tree (GiST) index on the geometry column and the raster datasets are loaded with the default raster constraints, and includes a GiST index on the bounding boxes of raster tiles.

SciDB does not support vector datatypes and uses array dimensions as proxies for indices. Conducting zonal analyses in SciDB requires we utilize an intermediary script to perform raster zonal analysis (pseudo code is found in Figure 2). First the Geographic Data Abstraction Library (GDAL) and OGR Simple Features Library (OGR) are used to access the geographic boundary and rasterize it. Next the SciDB-Py module is used to transfer the raster array into SciDB, where a raster join query, written in SciDB’s Array Functional Language (AFL), is submitted to perform the analysis. Since SciDB, does not use indices Figure 2 depicts the process of aligning the array dimensions of the masked raster and clipped global raster.

Our research sought, in part, to determine how raster partitioning, termed a tile size (PostgreSQL) or chunk size (SciDB), affects raster zonal analyses. Raster datasets are loaded into both platforms using a variety of partitioning sizes. The proposed optimal chunk sizes for SciDB (Table 3) are in the range of between five and 50 megabytes (Stonebraker et al., 2011). Chunk size is determined by estimating the number of megabytes based on the input value datatype, which is a 16 bit integer in this case. The raster dataset being loaded is a dense raster dataset and we employ a square non-overlapping chunk structure. There is no such stated ideal tile size for PostgreSQL. To account for this lack of an *a priori* tile size, the raster dataset was loaded into the database at various tile sizes, the results are discussed further.

### 3.2 Datasets Analyzed

One raster dataset and three different vector datasets were used for this comparison (Figure 3). The global raster used for comparison is Global Landcover 2000 from the European Space Agency, which has a spatial resolution of 0.0089 decimal degree or approximately one square kilometer at the equator. It is a dataset commonly employed by domains ranging from economics to ecology (Bartholomé & Belward, 2005) (Figure 3C). The vector datasets used for this analysis come from the US Census Cartographic boundaries (United States Census, 2016). Both raster and vector datasets were modified to use only the geospatial extent of the continental United States. The first vector dataset is the counties boundaries of the continental United States, which has approximately 3,000 features. The second dataset are the state boundaries of the continental United States, which have the same spatial extent as the first dataset but with only 49 geographic units. Finally, we increased the spatial scale further with the Large-Scale International Boundaries (LSIB), which is the merger of the EurasiaOceania and AfricaAmericas 2015 boundaries (United States Department of State, 2015). The merged dataset was modified by removing Antarctica and contains 249 country national boundaries. The LSIB dataset was chosen as it provides an accurate description of all current geospatial administrative boundaries for the globe. This will allow us to assess the capabilities and performance of both platforms when varying the partitioning strategies (termed tile size in PostgreSQL and chunk size in SciDB) for a global geospatial analysis.

## 4. Results

To directly compare platforms, we loaded each raster dataset into the database platforms at various partitioning sizes and then conducted zonal analysis on the each of the raster datasets. Figure 4 depicts the variation that occurs when conducting zonal analyses within PostgreSQL. Query performance is affected by tile size and the spatial irregularity of the vector datasets. Figure 4 captures vector irregularity in that it shows how the tile size chosen for one vector dataset with a high number of geographic features is unlikely to be beneficial for another with a small number of geographic features with the same extent. Therefore spatial irregularity affects the performance of zonal analysis for PostgreSQL.

Figure 4 depicts a “U-shaped” curve for the us_counties datasets, with an optimal tile size of 300×300 for this dataset. The results show that the us_counties dataset is strongly negatively affected by both large and small tile sizes. The strength of the adverse performance is due to high irregularity of the dataset and the large number of geographic features (3000). The us_counties dataset is characterized by a large number of geographic features with small spatial extents.

Figure 4 depicts a "L-shaped" performance curve for the us_states datasets. This dataset's poorest performance is when the tile size is the smallest, 50×50. As the tile size increases performance improves, and the ideal performance time occurs when the each feature size is roughly similar to raster tiles size. Performance degrades when geographic features intersect many smaller raster tiles.

This analysis demonstrates the complexity of choosing a single tile size for analyzing both vector datasets in PostgreSQL. Query performance is related to a two factors: tile size and feature irregularity, which is composed of the feature spatial extents and the number of features in the dataset. Both datasets show improved query performance time when the size of the raster tile is similar to the size of the geographic features. This reduces the time required for creating the masked features. As the tile size changes, from the ideal tile size, query time increases because in either case we are now creating more or larger arrays.

Figure 4 as initially presented suggests that continuously increasing the tile size would continue benefit the us_states dataset. With extended testing, however, we are able to reject that claim. Figure 5 depicts that continuing to increase the tile size will degrade the performance the us_states dataset. With us_states, the best query performance occurred with a tile size of 1000×1000, and after this query times began to increase. The effect of this trend is not as pronounced as with the us_counties dataset because the number of features is small. Still we conclude that tile size affects the performance of raster/vector overlays in PostgreSQL.

Figure 6 displays the averaged performance times of SciDB, which includes the time necessary to rasterize the vector dataset and transfer it to SciDB. As discussed earlier, SciDB only supports an array data model so an intermediary script is used to convert the vector datasets into an array and load it into SciDB so the analysis can be performed. Results of zonal analysis in SciDB indicate virtually no changes in performance when analyzing datasets with different vector complexities. The times reported for us_counties and us_states are almost identical, because the SciDB's model operates on the partitioned arrays. Each SciDB instance is given a portion of the rasterized polygon and a portion of the global raster dataset. The times are similar because the amount of data being analyzed is similar. Interestingly our results show that defined partition size has very little effect on the performance of SciDB as all queries are finishing within 10 seconds of each other.

Figure 6 does illustrate improved performance that comes when the partition size is 1,000×1,000 for both datasets. This improvement is due to the type of query implemented in SciDB. The query was written to reflect as accurately as possible the PostgreSQL query, which would allow a direct comparison between the two platforms. The zonal analysis conducted with SciDB required subsets the global raster array so that it can align with dimension of the polygon mask. The function "subarray" creates a resulting array that is likely to have a default chunk size of 1,000×1,000. When the resulting raster subset and the rasterized overlay have the same chunk size we see increased performance. If sub setting or "cropping" is frequently performed by an application – as it is with IPUMS-Terra – then choosing default chunk size of 1,000×1,000 would yield better performance.

The two platforms, SciDB and PostgreSQL w/ PostGIS, showed significant differences when performing zonal overlays between vector and raster datasets with states and counties datasets (us_states and us_counties respectively). Conducting zonal analysis within PostgreSQL's entity relationship model resulted in a wider variation in query time performance. Conversely, SciDB's array database model, which operates on the partitioned raster, had a very impressive consistent performance. Neither spatial extent of the geography, the number of geographic features, nor the tile size affected the performance with SciDB.

The final analysis increases the scale of the overlay to a global extent with the national LSIB boundaries. Due to the magnitude of the datasets, we considered alternative query structures that could improve performance for the global analysis. However, for PostgreSQL, the vector to raster overlay join was superior to employing a raster to raster join and those times are reported. For SciDB, a global array to global array join was superior to performing a number of sub analyses over the dataset and those times are reported. SciDB's implementation required that the all of the LSIB boundaries were rasterized and written to a geotiff file, which was then loaded into SciDB as an array. Rasterizing the boundaries shapefile using the serial GDAL library took approximately 50 seconds. We have not included the rasterization time as part of query perform time for SciDB as this was considered part of the data loading process for creating a new raster dataset. Additionally there are parallelization techniques using R or Python that could be employed to greatly reduce this time.

The direct comparisons between the two platforms utilized three different partitioning sizes. Tile sizes of 250, 500, and 1000 were chosen for PostgreSQL and are termed small, medium, and large respectively. These tile sizes were chosen to capture the maximum variation of PostgreSQL's performance. SciDB's chunk sizes were again determined by calculating the estimated number of megabytes resulting in chunk sizes of 1000, 3163, and 4473.

Figure 7 illustrates the advantage of SciDB over PostgreSQL when analyzing large raster datasets. PostgreSQL's best performance occurred when performing zonal statistics with the largest tile size. The average query time for PostgreSQL was approximately 497.54 seconds, with the largest tile size for 1000. The performance of SciDB on the larger raster dataset exceeded expectations. Many of the SciDB queries finished in approximately 120 seconds, with the best performance of 113.55 seconds resulting from a chunk size of 4473. The results show a similar pattern as in the previous analysis in that chunk size had little effect when conducting a raster zonal analysis.

Table 4 depicts the performance gains of using SciDB over PostgreSQL on big raster datasets. SciDB was able to process the larger raster dataset more efficiently than PostgreSQL because of its parallel architecture and array design. The category comparisons are useful for conducting a direct comparison between architectures. PostgreSQL's performance changes drastically as a function of the tile size of the raster dataset; this variation is undesirable when scaling geospatial applications. For example, category "small" in Table 4 depicts PostgreSQL's worst time and SciDB's best. A direct comparison between these two times results in a 29 times performance speedup for SciDB. The category "large" in Table 4 depicts PostgreSQL's best time and SciDB's best and still SciDB shows increased performance, at a little over one speedup per core. Clearly SciDB is a useful tool for analyzing big geospatial data.

## 5. Conclusion

We are in the era of big data. Much of the geospatial data currently collected already exemplifies big data and with the deployment of sensors and wearable devices, big spatial data will increasingly become the norm. One important challenge for GIScience is to develop accessible geospatial platforms that harness and provide access to geospatial data. The GIScience community has taken tremendous strides to develop efficient ways of parallelizing big spatial data for analysis, but still faces critical gaps in connecting data storage infrastructure to data processing infrastructure. IPUMS-Terra bridges this gap by providing both capabilities in a single platform.

Another challenge is that the tools used for working with raster and vector data are largely separate and this reduces the ability for the larger science community to fully utilize geospatial data. Geoprocessing services are a promising solution for raster and vector data integration as they allow for the integration of heterogeneous data types. However, many geoprocessing services are integrated within Web GIS platforms such as MapServer or GeoServer. These software systems are likely to be ineffective on big data as their primary purpose is data sharing and data visualization. Their geoprocessing algorithms only operate on data within their framework and must move data from external sources into their domain.

As spatial data infrastructure continues to grow, the gap between big data availability and high performance computation becomes a critical infrastructure challenge. IPUMS-Terra bridges this gap, because it contains large volumes of free data tethered to high-performance geospatial computation. Its users aren't required to analyze big geo data on a desktop GIS, but instead utilize the high performance computation (HPC) capabilities of the infrastructure. IPUMS-Terra is a critical piece of the spatial data infrastructure that addresses many challenges by supporting heterogeneous data types and linking users to big data computation resources. The project also demonstrates a potential evolution of web processing services, in which the analyses are pushed to the data instead of data being carried to analysis. Big data platforms and tools are likely to scale well for big spatial data and offer greater flexibility for end users.

This paper compares differing architectures and their effectiveness on big geospatial data. SciDB's partitioned array model is promising for analyzing big raster datasets as it is more efficient at scaling out analyses. However, its increased performance is only likely to be seen in queries that have large geographic extents that surpass the current limits of PostgreSQL. The raster dataset utilized in our experiments has a very coarse resolution and one of the first steps in future research will be to determine performance over a range of higher resolution datasets. Lastly, we plan to integrate SciDB into the IPUMS-Terra's ecosystem to support scalable and efficient raster computation.

## Acknowledgements

This research was supported by the National Science Foundation award OCI-0940818 and with support the National Institutes of Health supported Minnesota Population Center (R24 HD041023). We would also like to thank the members of IPUMS-Terra team who assisted with this research and providing guidance throughout this process. We would also like to thank our three anonymous reviewers who provided useful comments, which were able to incorporate into this manuscript.

## References

Armstrong MP (2000). Geography and computational science. Annals of the Association of American Geographers, 90(1), 146–156.

Babcock C (2013, 12 30). How DigitalGlobe Handles 2 Petabytes Of Satellite Data Yearly. Retrieved October 1, 2016, from http://www.networkcomputing.com/big-data/how-digitalglobe-handles-2-petabytes-satellite-data-yearly/767935503

Bartholomé E, & Belward A (2005). GLC2000: a new approach to global land cover mapping from Earth observation data. International Journal of Remote Sensing, 26(9), 1959–1977.

Batie SS (2008). Wicked problems and applied economics. American Journal of Agricultural Economics, 90(5), 1176–1191.

Brown VA, Harris JA, & Russell JY (2010). Tackling wicked problems through the transdisciplinary imagination. Earthscan.

CyberGIS Gateway. (2016). Retrieved November 1, 2016, from http://sandbox.cigi.illinois.edu/home/

Ding Y, & Densham PJ (1996). Spatial strategies for parallel spatial modelling. International Journal of Geographical Information Systems, 10(6), 669–698.

Dobson JE (1983). Automated Geography. The Professional Geographer, 35(2), 135–143. 10.1111/j.0033-0124.1983.00135.x

Eldawy A, & Mokbel MF (2015). The era of big spatial data In Data Engineering Workshops (ICDEW), 2015 31st IEEE International Conference on (pp. 42–49). IEEE.

Geospatial Data Analysis Building Blocks. (2016). Retrieved September 15, 2016, from https://purr.purdue.edu/projects/geodibbs

Google Earth Engine. (2016). Retrieved July 1, 2016, from https://earthengine.google.com/

Güting RH (1994). An introduction to spatial database systems. The VLDB Journal—The International Journal on Very Large Data Bases, 3(4), 357–399.

Hampton SE, Strasser CA, Tewksbury JJ, Gram WK, Budden AE, Batcheller AL, … Porter JH (2013). Big data and the future of ecology. Frontiers in Ecology and the Environment, 11(3), 156–162.

Haynes D, Ray S, Manson SM, & Soni A (2015). High performance analysis of big spatial data In Big Data (Big Data), 2015 IEEE International Conference on (pp. 1953–1957). IEEE.

Hofer B (2015). Uses of online geoprocessing technology in analyses and case studies: a systematic analysis of literature. International Journal of Digital Earth, 8(11), 901–917. 10.1080/17538947.2014.962632

Jaeger E, Altintas I, Zhang J, Ludäscher B, Pennington D, & Michener W (2005). A Scientific Workflow Approach to Distributed Geospatial Data Processing using Web Services In SSDBM (Vol. 3, pp. 87–90). Citeseer.

Jarvis A, Reuter HI, Nelson A, Guevara E, & others. (2008). Hole-filled SRTM for the globe Version 4. Available from the CGIAR-CSI SRTM 90m Database (Http://Srtm. Csi. Cgiar. Org).

Mayer-Schönberger V, & Cukier K (2013). Big data: A revolution that will transform how we live, work, and think. Houghton Mifflin Harcourt.

Murdoch TB, & Detsky AS (2013). The inevitable application of big data to health care. Jama, 309(13), 1351–1352. [PubMed: 23549579]

Olasz A, Thai BN, & Kristóf D (2016). A New Initiative for Tiling, Stitching and Processing Geospatial Big data in Distributed Computing Environments. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences, 111–118.

O'Neill BC, MacKellar FL, & Lutz W (2005). Population and climate change. Cambridge University Press.

Papadias D, Kalnis P, Zhang J, & Tao Y (2001). Efficient OLAP Operations in Spatial Data Warehouses In Jensen CS, Schneider M, Seeger B, & Tsotras VJ (Eds.), Advances in Spatial and Temporal Databases: 7th International Symposium, SSTD 2001 Redondo Beach, CA, USA, July 12–15, 2001 Proceedings (pp. 443–459). Berlin, Heidelberg: Springer Berlin Heidelberg Retrieved from 10.1007/3-540-47724-1_23

Pavlo A, Paulson E, Rasin A, Abadi DJ, DeWitt DJ, Madden S, & Stonebraker M (2009). A Comparison of Approaches to Large-scale Data Analysis In Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data (pp. 165–178). New York, NY, USA: ACM 10.1145/1559845.1559865

Planthaber G, Stonebraker M, & Frew J (2012). EarthDB: scalable analysis of MODIS data using SciDB In Proceedings of the 1st ACM SIGSPATIAL International Workshop on Analytics for Big Geospatial Data (pp. 11–19). ACM.

Ramsey P (2015, 10 7). Retrieved June 1, 2016, from http://postgis.net/2015/10/07/postgis-2.2.0/

Ray S, Simion B, Brown AD, & Johnson R (2014). Skew-resistant parallel in-memory spatial join In Proceedings of the 26th International Conference on Scientific and Statistical Database Management (p. 6). ACM.

Stonebraker M, Brown P, Poliakov A, & Raman S (2011). The architecture of SciDB In International Conference on Scientific and Statistical Database Management (pp. 1–16). Springer.

Taylor KE, Stouffer RJ, & Meehl GA (2012). An overview of CMIP5 and the experiment design. Bulletin of the American Meteorological Society, 93(4), 485.

United States Census. (2016, 4 28). Retrieved December 1, 2016, from https://www.census.gov/geo/maps-data/data/tiger-cart-boundary.html

United States Department of State. (2015, 1 23). Retrieved June 1, 2016, from https://catalog.data.gov/dataset

United States Geological Survey. (2016). Retrieved August 2, 2016, from https://nationalmap.gov/

Vagata P, & Wilfong K (2014, 4 10). Scaling the Facebook data warehouse to 300 PB. Retrieved June 1, 2016, from https://code.facebook.com/posts/229861827208629/scaling-the-facebook-data-warehouse-to-300-pb/

van Rijmenam M (2016, 7 18). Self-driving Cars Will Create 2 Petabytes Of Data, What Are The Big Data Opportunities For The Car Industry? Retrieved November 15, 2016, from https://datafloq.com/read/self-driving-cars-create-2-petabytes-data-annually/172

Wang S (2010). A CyberGIS Framework for the Synthesis of Cyberinfrastructure, GIS, and Spatial Analysis. Annals of the Association of American Geographers, 100(3), 535–557. 10.1080/00045601003791243

World Bank. (2012). World Development Indicators 2012. World Bank Publications.

Wright DJ, & Wang S (2011). The emergence of spatial cyberinfrastructure. Proceedings of the National Academy of Sciences, 108(14), 5488–5491. 10.1073/pnas.1103051108

Yang C, Goodchild M, Huang Q, Nebert D, Raskin R, Xu Y, … Fay D (2011). Spatial cloud computing: how can the geospatial sciences use and help shape cloud computing? International Journal of Digital Earth, 4(4), 305–329. 10.1080/17538947.2011.587547

Yang CP, & Raskin R (2010). Geospatial cyberinfrastructure (GCI). Computers, Environment and Urban Systems, 34(4), 263 10.1016/j.compenvurbsys.2010.05.004

Yang C, & Raskin R (2009). Introduction to distributed geographic information processing research. International Journal of Geographical Information Science, 23(5), 553–560. 10.1080/13658810902733682

Yue P, Zhou H, Gong J, & Hu L (2013). Geoprocessing in Cloud Computing platforms – a comparative analysis. International Journal of Digital Earth, 6(4), 404–425. 10.1080/17538947.2012.748847

Zhang T, & Tsou M-H (2009). Developing a grid-enabled spatial Web portal for Internet GIServices and geospatial cyberinfrastructure. International Journal of Geographical Information Science, 23(5), 605–630. 10.1080/13658810802698571

Zhou X, Abel DJ, & Truffet D (1998). Data Partitioning for Parallel Spatial Join Processing. GeoInformatica, 2(2), 175–204. 10.1023/A:1009755931056

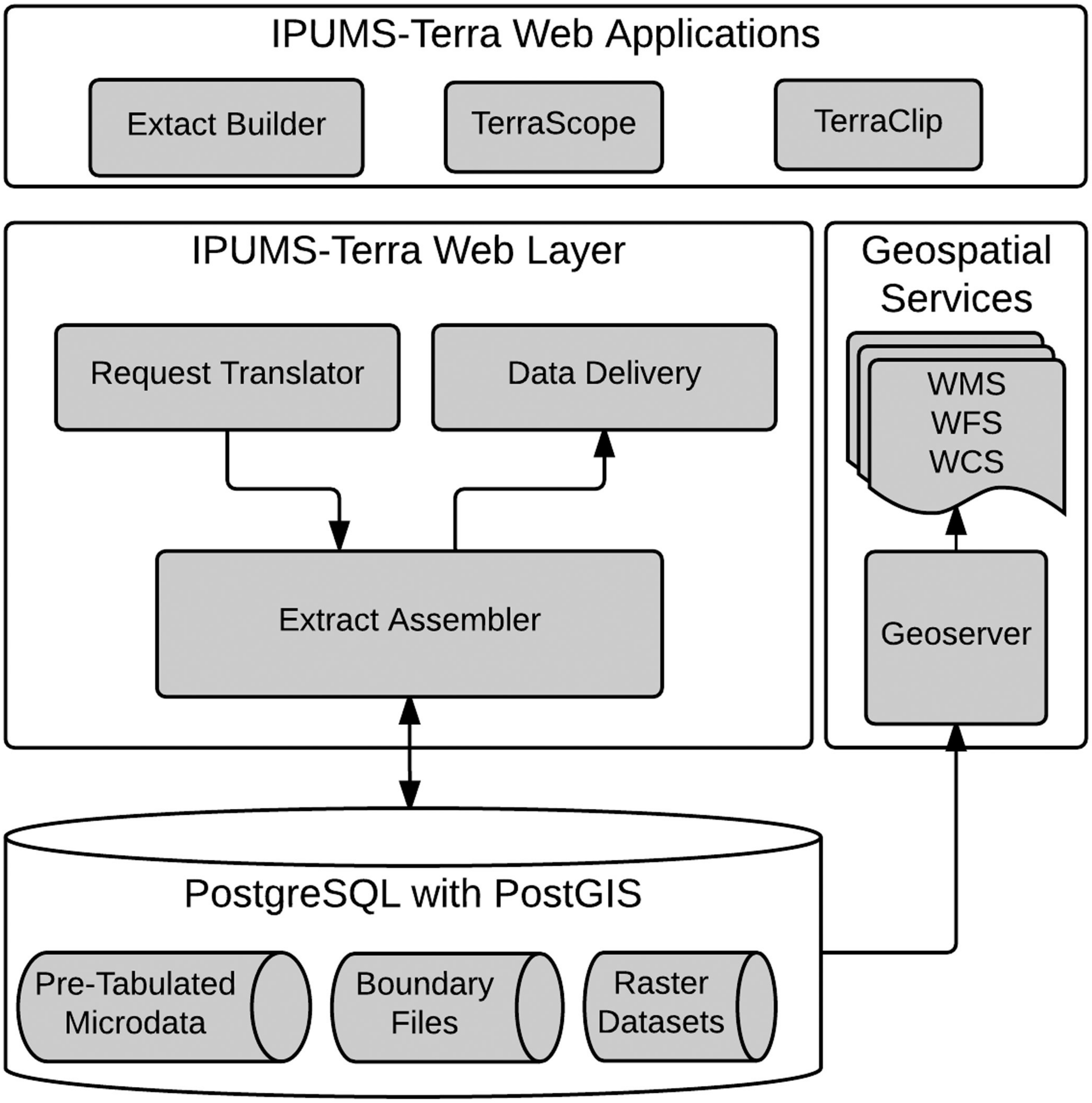

**Figure 1.**
IPUMS-Terra System Architecture

```
#Connect to SciDB and Open Existing Files
scidb= scidbpy.connect()
raster = gdal.Open(rasterfile)
boundary = ogr.Open(boundaryfile)

#Convert Polyon to Raster Mask
rasterizedBoundary = RasterizePolygon(raster, boundary)
#Transfer Array to SciDB
scidb.from_array(rasterizedBoundary)
#Perform Zonal Analysis
sdb.query( " grouped_aggregate(join({rasterizedBoundary}, subarray( glc2000{chunk}, minX, minY, maxX, MaxY)) ,  mean(value), count(value), id) ")
```

**Figure 2.**
SciDB Raster Zonal Analysis Pseudocode

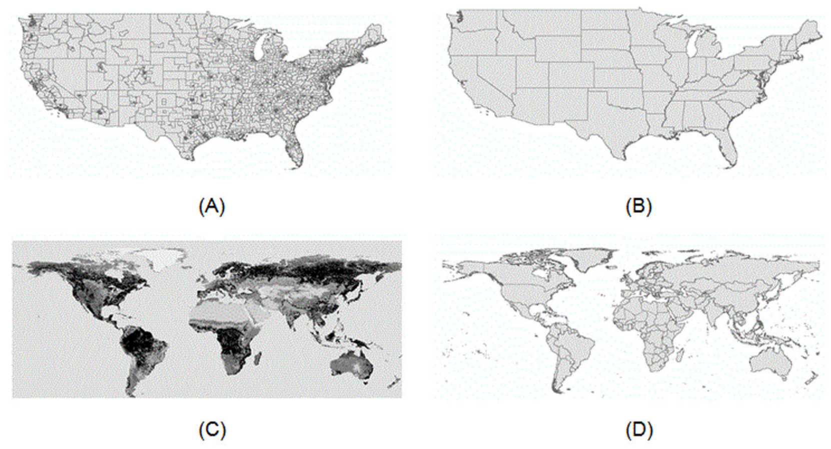

**Figure 3.**
Datasets Used for Scaling Zonal Statistics

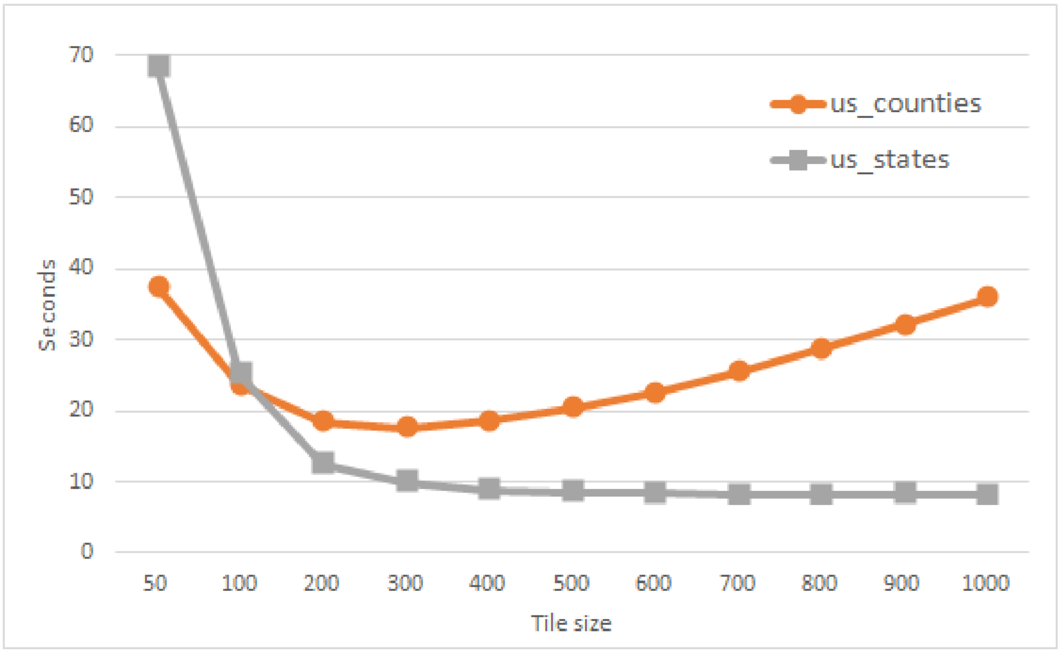

**Figure 4.**
Performance of PostgreSQL Zonal Analysis Based on Raster Tile Size

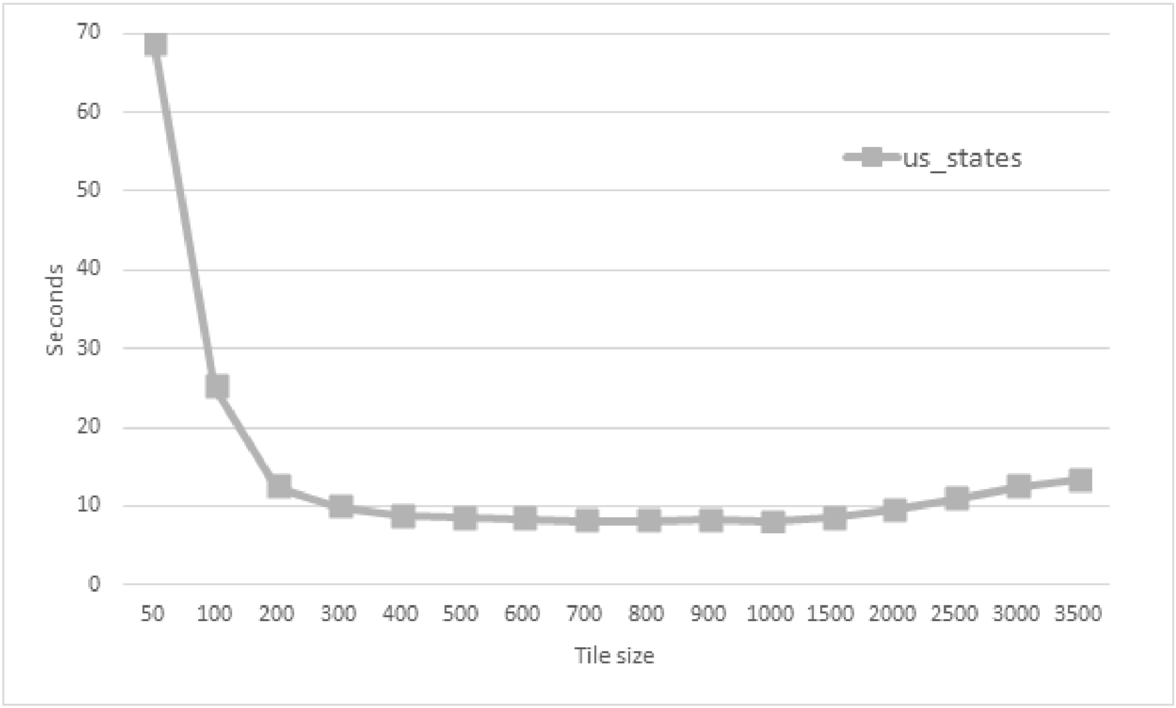

**Figure 5.**
Extended Analysis of PostgreSQL Zonal Statistics for US States Dataset

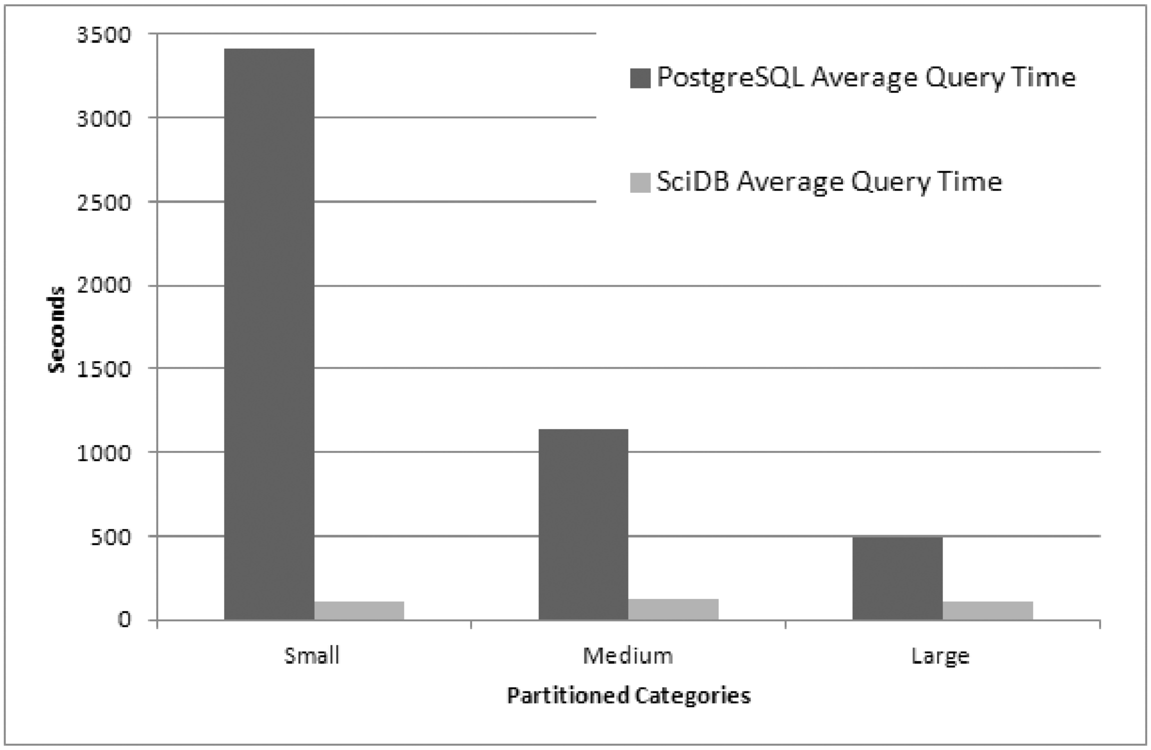

**Figure 6.**
Performance of SciDB for Zonal Analysis Based on Array Chunk Size

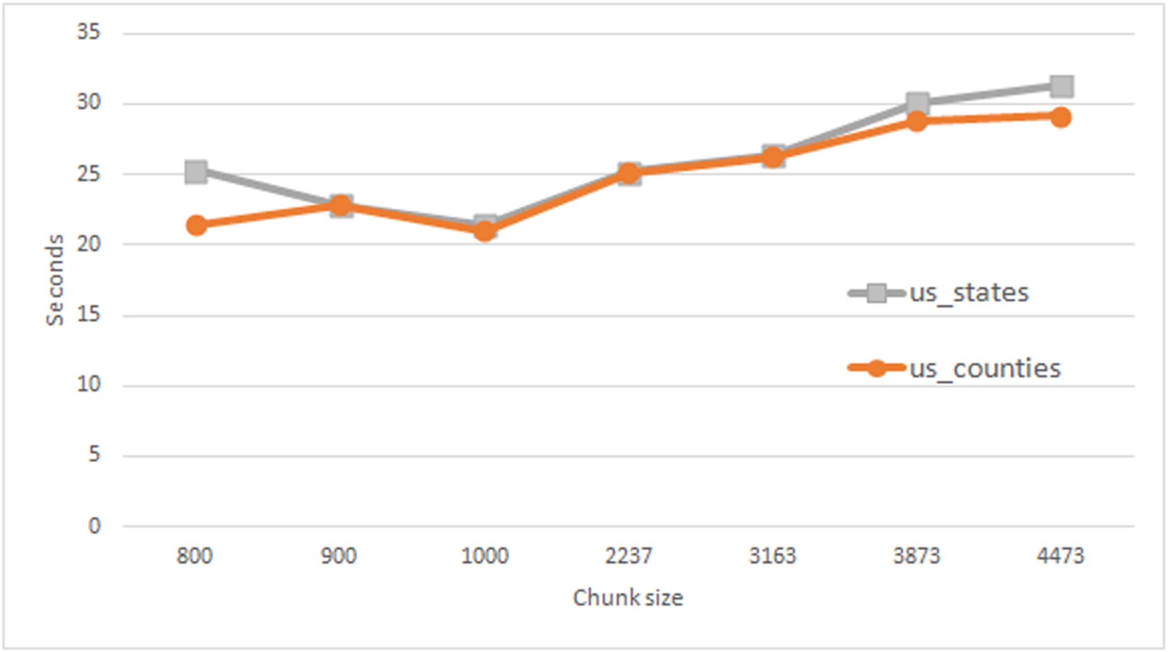

**Figure 7.**
Comparison of Large Scale Zonal Statistics

**Table 1.**

Characteristics of High Performance Geocomputation Environments

| | Vector Analysis | Raster Analysis | Scalable |
|---|---|---|---|
| Spatial Hadoop | X | | X |
| GeoMesa | X | | X |
| GeoTrellis | | X | X |
| PostgreSQL w/ PostGIS | X | X | |
| Greenplum DB | X | | X |
| Rasdaman | | X | X |
| SciDB | | X | X |

**Table 2.**

IPUMS-Terra Integrated Data Structures

| **Data Structures** | **Spatial representations** |
|---|---|
| *Microdata*: Each record represents an individual person, household, or firm | Each record includes a code identifying a geographic polygon (e.g., state) or a point (street address) |
| *Vector*: Each record contains information about a place, such as census tabulations. | Each record includes coordinates defining a geographic polygon or point (e.g., zip code, environmental zone) |
| *Raster*: Two-dimensional grid of pixels, where each pixel provides a value for a variable | Each grid cell corresponds to a rectangular area on the ground |

**Table 3.**

Chunk sizes for SciDB

| Estimated MB | Chunk dimensions | Pixels per chunk | Estimated number of chunks |
|---:|---:|---:|---:|
| 1 | 1000 * 1000 | 1,000,000 | 659.3 |
| 5 | 2237 * 2237 | 5,004,169 | 131.8 |
| 10 | 3163 * 3163 | 10,004,569 | 65.9 |
| 15 | 3873 * 3873 | 15,000,129 | 43.9 |
| 20 | 4473 * 4473 | 20,007,729 | 32.9 |

**Table 4.**

SciDB Performance Speedup

| | PostgreSQL Average Query Time | SciDB Average Query Time | Speedup | Speedup per Core |
|---|---|---|---|---|
| **Small** | 3415.808 | 115.750 | 29.510 | 7.378 |
| **Medium** | 1146.386 | 121.930 | 9.402 | 2.350 |
| **Large** | 497.549 | 113.552 | 4.382 | 1.095 |