https://doi.org/10.5194/egusphere-2025-1558
Preprint. Discussion started: 21 May 2025

Configuring parallel use of custom ArcGIS toolboxes in a Linux high-performance computing environment

Jeremy Baynes[1], Jacob Tafrate[2], Donald Ebert[1], Steven Lennartz[3]

[1] U.S. Environmental Protection Agency (EPA), Office of Research and Development (ORD), Center for Public Health and Environmental Assessment (CPHEA), 109 T.W. Alexander Drive, Research Triangle Park, NC 27711, USA
[2] Oak Ridge Associated Universities supporting U.S. Environmental Protection Agency (EPA), Center for Public Health and Environmental Assessment (CPHEA), 109 T.W. Alexander Drive, Research Triangle Park, NC 27711, USA
[3] U.S. Environmental Protection Agency (EPA), Office of Mission Support (OMS), Office of Information Management (OIM), 109 T.W. Alexander Drive, Research Triangle Park, NC 27711, USA

Correspondence to: Jeremy Baynes (baynes.Jeremy@epa.gov)

Abstract. High performance computing (HPC) clusters offer an abundance of processors, memory, and data storage that can help meet growing computational demands for large geospatial analyses. Esri's proprietary ArcGIS toolboxes are a common way to develop, document, and share geospatial code; however, these toolboxes are typically developed for use in Esri's ArcGIS Pro for Windows which limits their use in Linux, the dominant HPC operating system. Esri's support of Linux for their server software opens the possibility of using ArcGIS toolboxes in an HPC environment, but use of this software proved problematic for concurrent, parallel processing offering little benefit over a single workstation. We developed a solution using container technology allowing us to run hundreds of concurrent tasks that use ArcGIS toolboxes. This approach is designed to take advantage of the many existing ArcGIS toolboxes and use them in parallel for workflows that can be reasonably divided into independent sub-tasks. With careful setup and use of an HPC job scheduler, we reduced total processing time for one analysis 95.6 % by dividing the analysis into 202 sub-tasks. This solution allows other HPC users to take advantage of parallel use of ArcGIS Toolboxes, avoid the effort of translating existing workflows to Linux native software, and establish a bridge between desktop GIS and HPC systems.

1 Introduction

Geographic information systems (GIS) are powerful information technology tools that help us model and better understand ecosystems, human health, urban planning, disaster response, and numerous other disciplines that inherently rely on spatial data. As a procedure-oriented software, GIS modeling often relies on processing results from one analysis as inputs to subsequent analyses (Zhu et al., 2021). The number, complexity, and iterative nature of these connected steps makes GIS workflows well suited for automation with software code, and sharing those workflows are common (Müller et al., 2013;Scheider et al., 2019). As those workflows are used to analyze data provided in ever increasing frequency, resolution, and spatial coverage, required computing resources can be stretched beyond the limits of even high-end desktop workstations.

https://doi.org/10.5194/egusphere-2025-1558
Preprint. Discussion started: 21 May 2025

To meet the growing computational demands of geospatial workflows, high performance computing (HPC) offers a potential solution.

HPC, and similarly high-throughput computing (HTC), offer an abundance of processors, memory, and storage in a connected environment. Generally, HTC is a collection of computers performing predefined, independent sub-tasks whereas HPC is an interconnected series of compute nodes that allow for communication between processors and can effectively act as a single large computer (Fienen and Hunt, 2015;Leonard et al., 2014). It is common to run complex scenarios in an HPC environment using hundreds or thousands of cores using custom parallel processing code. Effectively using an HPC environment often requires expertise in both a specific domain (e.g., climatology, fluid dynamics, hydrology) and distributed computing (Barrios-Hernandez et al., 2024;Raj et al., 2020). Alternatively, a typical workflow for HTC is to divide a large problem into smaller, independent sub-tasks, process each sub-task concurrently on a unique computer, and then use post-processing to merge results; a workflow often referred to as divide-and-conquer or embarrassingly parallel (Erickson et al., 2018). In the most basic terms, the start and completion times of HTC sub-tasks do not matter whereas HPC sub-tasks are often interdependent on the completion of other sub-tasks. However, using an HPC job scheduler, optimally organizing input data, and isolating workflows can allow an HPC system to function as an HTC system. For example, an HPC system with 64 nodes, each with 32 cores, could run a single 2,048 core job, 2,048 simultaneous single core jobs, or any combination of cores per job using no more than 2,048 cores. Huerta et al. (2019) encouraged the use of HTC workloads in HPC environments to take advantage of otherwise unused compute nodes and to further the interoperability of HPC and HTC environments in future planning.

There is a long history of advancing geospatial efforts with parallel processing and HPC (Clematis et al., 2003;Healey et al., 2020;Openshaw and Turton, 2005;Tang and Wang, 2020), but there is no single high-performance GIS software. HPC users rely on developing new software or using existing software in creative ways to advance their efforts. GDAL, Grass GIS, QGIS, R, and Python are all examples of mature, open-source software and libraries heavily used in the GIS community (Coetzee et al., 2020). However, there is an obvious market leader in GIS software. A 2015 press release proclaimed that Esri, a commercial GIS software company, accounted for 43 percent of the GIS market (Esri, 2015). A 2018 survey that targeted geospatial software experts (n=454) reported that 66% of respondents listed Esri products as the geospatial software they used (Vandewalle et al., 2020). To create and share reproducible workflows, Esri has long provided their users the ability to develop custom code with their suite of geospatial tools. From ArcMap 9.0 released in 2004, Esri has made use of the Python programming language in their software and provided programmatic access to their geoprocessing tools with their custom Python site package, ArcPy (Esri, 2025b). Esri also provides users the ability to develop, document, and share their workflows with other Esri users through custom, user-friendly ArcGIS toolboxes. ArcGIS toolboxes can be developed using Python or Esri's visual programming language, ModelBuilder (Esri, 2025c).

https://doi.org/10.5194/egusphere-2025-1558
Preprint. Discussion started: 21 May 2025

Domain experts develop and share ArcGIS toolboxes across disciplines and industries. A cursory search found recent efforts highlighting new ArcGIS toolboxes for monitoring harmful algal blooms (Saltus et al., 2022), creating toxicological index visualizations (Fleming et al., 2022), estimating soil erosion caused by bridge construction (Ahmari et al., 2022), measuring health care accessibility (Hashtarkhani et al., 2024), and generalizing space-time prisms for moving objects (Loraamm et al., 2020). Because of the wide use of Esri's software, there is an abundance of publicly available and an uncountable number of privately held ArcGIS toolboxes across a wide spectrum of domains. ArcGIS toolboxes are typically developed in and designed for use in Esri's desktop software, currently ArcGIS Pro, which requires a Microsoft Windows operating system. This type of use is limited by the processing capabilities of a user's individual workstation.

There has been some success in using Esri technology in an HPC or HTC environment, particularly with Microsoft Windows-based operating systems. In Windows-based HPC environments, Esri technology has been used to generate high resolution digital elevation models (Zheng et al., 2018), model land use land cover change (Pijanowski et al., 2014), and process climate data (Wang et al., 2013). Others have used Esri technology in Windows-based HTC environments to process LiDAR (Zurqani et al., 2020), detect wetlands (Leonard et al., 2012) and extract drainage networks (Gong and Xie, 2009). Leonard et al. (2014) found an almost 100-fold reduction in processing time running Esri Python code with 132 Windows based workstations in an HTC cluster. Of note, they found greater performance improvements using open-source software in an HPC environment but did not test Esri performance in an HPC environment because of limitations in Esri products of the time to rely on single processors and Windows based systems (Leonard et al., 2014).

There are far fewer documented examples of using Esri software in a Linux-based HPC or HTC environment. Linux, an open-source operating system, is the predominant HPC operating system family. Since May 2015, a biannual list of the top 500 HPC systems have exclusively been Linux based operating systems (Top500, 2024). Wang et al. (2006) developed a graphical GIS using Esri's ArcGIS Engine for use in a Linux HPC environment, but ArcGIS Engine has entered mature support and will soon be retired. Tang and Matyas (2018) described a cross platform library for using ArcGIS on a Linux HPC cluster, but it relied on porting ArcGIS code to a Windows server. Despite the many ArcGIS Toolboxes and predominance of Linux HPC environments, it is difficult to find examples of the two used together. A recently updated Esri community message board about using ArcGIS and HPC recommends using ArcGIS Enterprise or Server for a Linux based cluster but provides little guidance on how to accomplish this (GeriMiller, 2023). This is likely due to the complex and unique configurations of each HPC system and installation in a HPC environment is not an officially supported Esri solution.

While relying on proprietary software can hinder the reproducibility of research (Trisovic et al., 2022) and limit imaginative solutions (Clematis et al., 2003), ArcGIS toolboxes are popular, open (i.e., they rely on Python code), often well documented and vetted, and applicable in diverse domains. Here, we describe our efforts to deploy multiple ArcGIS toolboxes in parallel in a Linux HPC environment. If practical and transferable, this solution would allow HPC users to take advantage of parallel

https://doi.org/10.5194/egusphere-2025-1558
Preprint. Discussion started: 21 May 2025

use of ArcGIS Toolboxes, avoid the effort of translating established workflows to Linux native software, and provide a bridge between classic, desktop GIS and high-end scientific computing.

2 Materials and methods

2.1 ArcGIS toolbox

The United States Environmental Protection Agency's (EPA) Office of Research and Development (ORD) developed and maintains the Analytical Tools Interface for Landscape Assessments (ATtILA) for ArcGIS Pro (US EPA, 2023a). This custom ArcGIS toolbox calculates many landscape ecology and landscape characteristics metrics and summarizes results by a user provided reporting unit (e.g., watershed, census block, standard grid). ATtILA accepts a broad range of input data and has many adjustable parameters. ATtILA was originally released in 2004 for Esri ArcView (Ebert, 2004), updated to Esri ArcMap in 2016, and was recently updated for compatibility with ArcGIS Pro and Python 3 (US EPA, 2023b). In 2025, ATtILA released with two new tools and two new utilities along with additional enhancements and features. ATtILA has 25 tools and utilities organized into four categories and has been tested with ArcGIS Pro 3.3. The "SpatialAnalyst" extensions in ArcGIS Pro is required for ATtILA.

2.2 High performance computing

The EPA's HPC environment, named Atmos, has two independent compute clusters with different specifications. For this project, we selected the cluster containing two racks, each with 64 nodes. Each node has 16 dual-core Intel E5-2697A v4 processors and 256 GB RAM running Red Hat Enterprise Linux 8.10 (Ootpa) with BASH 4.4.20. This cluster has 100 Gbit/s EDR Infiniband interconnects and a shared GPFS file system with four petabytes of disk space. Atmos uses Simple Linux Utility Resource Management (Slurm), an open-source software, for scheduling, starting, and managing jobs on HPCs (Yoo et al., 2003) and Lmod 8.7.23 to manage HPC environment modules.

2.3 Container

Installing ArcGIS Server, using the packaged installer, and accessing the Python 3 runtime under a user profile was trivial on Atmos; however, direct installation proved problematic in initial testing when attempting to run multiple analyses in parallel. We possibly could have implemented a scheme using unique installations of ArcGIS Server multiple times (e.g., /arcgis/server_1, /arcgis/server_2, /arcgis/server_*n*) but this would be cumbersome, take unnecessary time, and use extensive disk space once *n* grew larger than a handful of installations.

Containers offer many appealing qualities for scientific data analysis by providing a reproducible and sharable environment in addition to isolation between applications (Abraham et al., 2020). For this effort, the isolated nature of containers was desirable as we wanted to use multiple instances of the same application independently on a single node. Using container technology

https://doi.org/10.5194/egusphere-2025-1558
Preprint. Discussion started: 21 May 2025

and individual scratch workspaces avoids the conflicts that arise from multiple uses of the same software on a single node. Scratch workspaces allow an immutable container to function as a read/write image.

Docker is a popular container technology (Abraham et al., 2020;Lin et al., 2020). In fact, a dormant GitHub repository for installing ArcGIS for Server on Docker from a senior Esri subject matter expert provided inspiration for our solution (Raad, 2016). While popular in cloud environments, Docker has several downsides for use in an HPC environment, notably a security concern of requiring root (i.e., administrator) permissions to run (Abraham et al., 2020). This is a non-negotiable barrier for many HPC environments, and as such, Docker is not supported on Atmos.

Apptainer (formerly Singularity) is a container technology with some overlap and compatibility with Docker that was developed with added security for use in HPC clusters (Kurtzer et al., 2017). Like other container technologies, Apptainer uses a definition file to build an immutable image (i.e., container), termed a Singularity Image Format (SIF) image. We have configured an SIF image with Rocky Linux OS version 8 and ArcGIS Enterprise 11.3 with the Python 3 runtime that supports ArcPy (US EPA, 2025). The SIF image was built outside of Atmos because root access is required to build a SIF image. One consideration for building a container with ArcGIS Server, is that ArcGIS Server does not support root level installation and must be installed under a user profile; therefore, the username, user ID, group, and group ID for the individual user of the SIF image on the HPC system are required when building the SIF image. This image is then limited to use by a single user and each user would need their own SIF image. A BASH script and alias were used to access ArcPy from the SIF image (US EPA, 2025).

2.4 Data

Several publicly available raster and vector datasets were used to test feasibility and performance of ATtILA on Atmos. Land cover data from the 2023 Annual National Land Cover Database (NLCD) (US Geological Survey, 2024) were used. Land cover is a gridded dataset (i.e., raster) where each cell represents the dominant biophysical land type in that area (e.g., forest, developed, grassland). These data are 8-bit integer GeoTiff, projected to WGS84 Albers Equal Area projection, and at 30 m resolution. There are approximately nine billion grid cells in a single year NLCD.

Hydrologic units from the National Hydrography Dataset (NHD) Plus version 2 Watershed Boundary Dataset (WBD) were used as boundaries and reporting units for our analyses. The WBD is a hierarchical system that identifies the 18 major drainage areas in the conterminous United States with a unique two-digit code and progressively appends a new two-digit code to uniquely identify smaller, nested hydrologic units within each of six levels (Fig. 1) (US Geological Survey). Two-digit and four-digit hydrologic unit codes (HUC-2, HUC-4), from the WBD were used to divide the analysis into manageable sub-tasks while twelve-digit hydrologic units (HUC-12) were used as the summary units (i.e., reporting unit) for all analyses (Table 1). HUC boundaries were projected to match the NLCD and stored in an Esri file geodatabase.

https://doi.org/10.5194/egusphere-2025-1558
Preprint. Discussion started: 21 May 2025

Figure 1: Watershed Boundary Dataset structure. From: US Geological Survey. Watershed Boundary Dataset. 20250204 https://www.usgs.gov/national-hydrography/watershed-boundary-dataset. Public domain.

Table 1. Watershed Boundary Dataset statistics for conterminous U.S.

Level	Count	Average Size (km2)	Std Dev (km2)
HUC-2	18	438,681	283,111
HUC-4	202	39,095	21,300
HUC-12	82,915	95.24	69.05

Hydrology data was acquired from the National Hydrography Dataset (NHD) Plus version 2 (McKay, 2012) by state. The NHD geodatabase was processed using the Process NHD for EnviroAtlas Analyses ATtILA utility to output streamlines and areas split by HUC-4s. Using ArcGIS Pro's merge function within a python script, lines and areas were combined into conterminous U.S. scale streamlines and stream areas. These datasets contain 18,752,256 lines and 2,050,871 areas. Data were projected to match the NLCD and added to the Esri geodatabase.

2.5 Workflow and benchmarks

The Riparian Land Cover Proportions (RLCP) tool within ATtILA calculates the percentage of each land cover type near water features within a reporting unit (US EPA, 2023b). This tool uses multiple spatial analyses to determine landscape types within

https://doi.org/10.5194/egusphere-2025-1558
Preprint. Discussion started: 21 May 2025

buffers around water features while ensuring areas from adjacent watersheds are excluded. Due to the complex analysis and large file size of input datasets, this tool has consistently failed to run at a national scale, even on larger Windows workstations, forcing us to divide the analysis into multiple sub-tasks. Historically, we have completed this analysis in series on a single workstation over several days; however, the independent nature of the analysis (e.g., areas outside of the reporting units are excluded) makes this tool an ideal candidate for running multiple sub-tasks in parallel.

A Python script was used to identify sub-tasks, create an output location, import the ATtILA toolbox, set parameters (Table A1), and initiate ATtILA calculation. Conterminous U.S. scale inputs are split by a sub-unit (i.e., HUC-2 or HUC-4), processed either in-series or in parallel, and finally output tables are merged, producing a result table at the conterminous U.S. scale (Fig. 2).

https://doi.org/10.5194/egusphere-2025-1558
Preprint. Discussion started: 21 May 2025

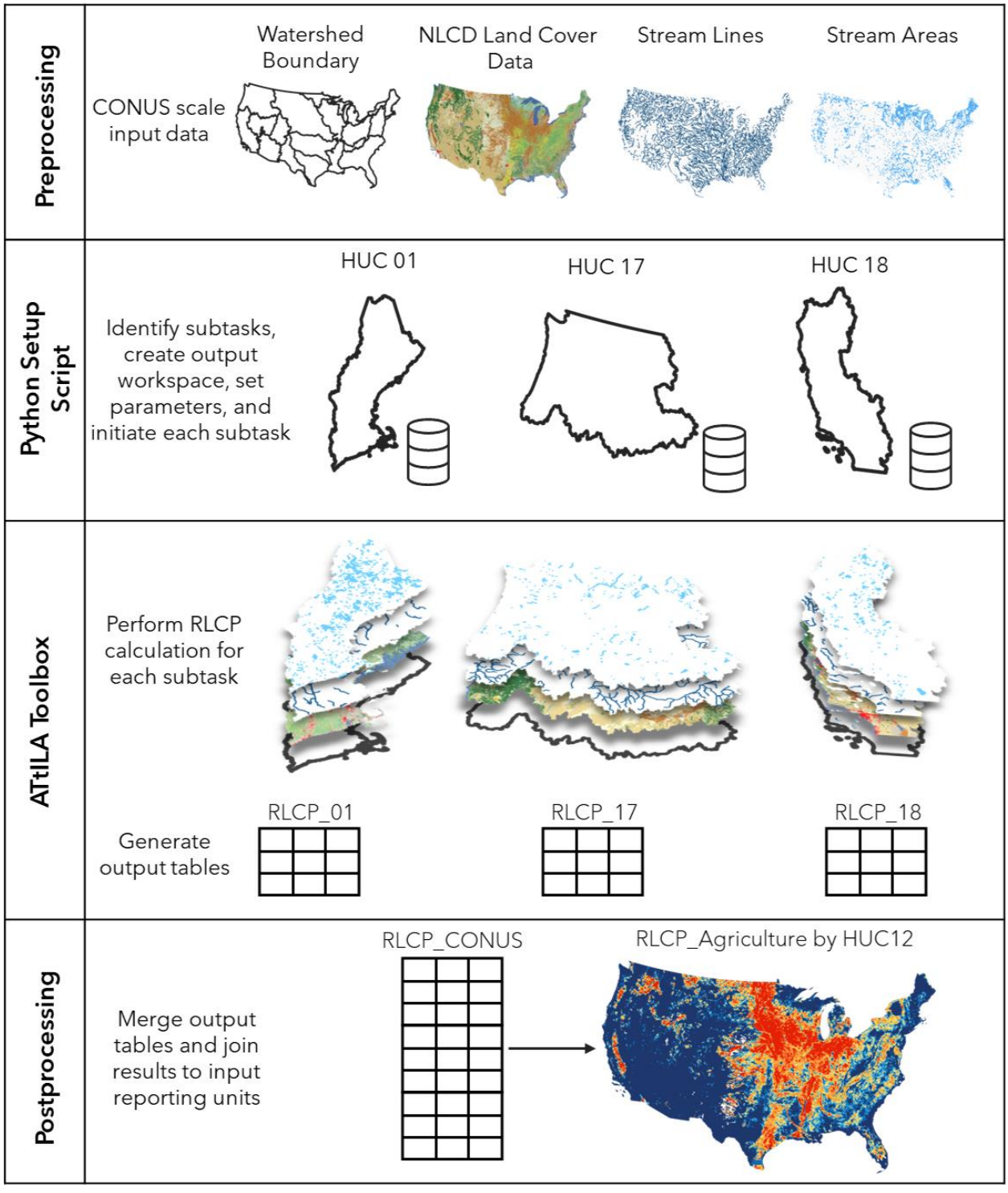

Figure 2: Processing steps for calculating Riparian Land cover proportions at the conterminous U.S. scale

https://doi.org/10.5194/egusphere-2025-1558
Preprint. Discussion started: 21 May 2025

Processing times between four configurations were tested and compared. Figure 3 illustrates workflows in both series and in parallel.

1. Atmos with a direct installation of ArcGIS Enterprise and sub-tasks processed in series.
2. Atmos with a direct installation of ArcGIS Enterprise and sub-tasks processed in parallel.
3. Atmos with ArcGIS Enterprise installed in an Apptainer SIF and sub-tasks processed in series.
4. Atmos with ArcGIS Enterprise installed in an Apptainer SIF and sub-tasks processed in parallel.

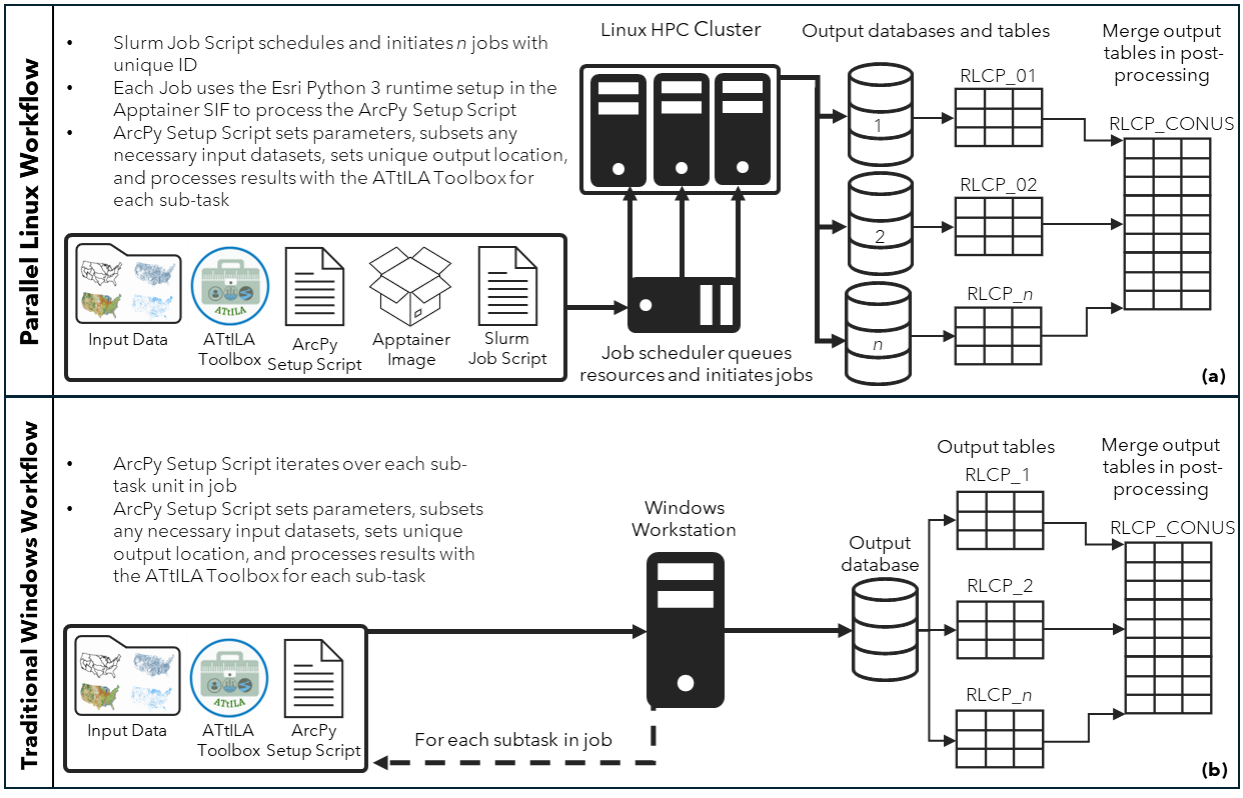

Figure 3: The parallel linux (a) and traditional Windows (b) workflows utilizing divide-and-conquer GIS processing

For series workflows, a Slurm script was used to request one job with four CPUs and 32 GB RAM. The Slurm script iterated through each sub-task unit ID (e.g., HUC-4 ID) and called a Python script with the unit ID as an argument. The Python script identified the sub-task boundary, set parameters and output locations, and processed RLCP with ATtILA for that sub-task unit. For parallel workflows, a Slurm job array was used to queue jobs on Atmos with the same specifications as the series workflow (Table A1). Instead of iterating sub-tasks through one job, a job array can initiate a job for each sub-task. Like the series workflow, each job in the array calls the same Python script with a unit ID as an argument. The Python script then identifies the sub-task boundary, set parameters and output locations, and processes RLCP with ATtILA. Parallel workflows were limited

https://doi.org/10.5194/egusphere-2025-1558
Preprint. Discussion started: 21 May 2025

to a maximum of 128 jobs at once (512 total cores). Setting a maximum number of jobs was not done because of a limitation of Atmos, but to be mindful of resources on a shared system. Slurm job arrays begin each job at the completion of any currently running job so that there is effectively always the maximum number of jobs processing.

For each of these configurations we tested two sub-unit scales to split the conterminous U.S. scale inputs into sub-tasks (i.e., HUC-2 and HUC-4). The same input files were used for all Atmos tests. We compared both individual processing time and total processing time for both sub-unit scales. Processing times were based on timestamps captured in ATtILA log files and therefore do not include a minimal amount of time for work completed in the Python setup script.

We use speedup (*S*) and parallel efficiency (*E*) to quantify the performance difference between in-series and parallel processing. Defined as the ratio of serial processing time (T_{series}) to parallel processing time ($T_{Parallel}$), speedup measures performance improvement associated with parallelization (Equation 1) (Eager et al., 1989). Parallel Efficiency, calculated as speedup divided by the number of processing units (*N*), evaluates how effectively computational resources are utilized during parallel processing (Equation 2) (Eager et al., 1989).

$$S = \frac{T_{series}}{T_{parallel}} \quad (1)$$

$$E = \frac{S}{N} \quad (2)$$

3 Results

3.1 Processing in series

Total processing time when accessing ArcGIS Python Runtime with an Apptainer SIF image was 3,435.7 minutes with HUC-2 sub-tasks and 3,473.5 minutes with HUC-4 sub-tasks. Individual sub-task processing ranged from 31.9 - 493.3 minutes with an average of 190.1 minutes (σ = 134.8) with HUC-2 sub-tasks and 1.8 – 135.2 minutes with an average of 16.47 minutes (σ = 15.7) with HUC-4 sub-tasks (Fig. 4).

Total processing time in series with a direct installation of ArcGIS Server on Atmos was 3,390.9 minutes with HUC-2 sub-tasks and 3,700.5 minutes with HUC-4 sub-tasks. Individual sub-tasks processing ranged from 32.3 – 496.5 minutes with an average of 187.5 minutes (σ = 134.1) with HUC-2 sub-tasks and 2.0 – 143.3 minutes with an average of 17.3 minutes (σ = 16.6) with HUC-4 sub-tasks (Fig. 4).

https://doi.org/10.5194/egusphere-2025-1558
Preprint. Discussion started: 21 May 2025

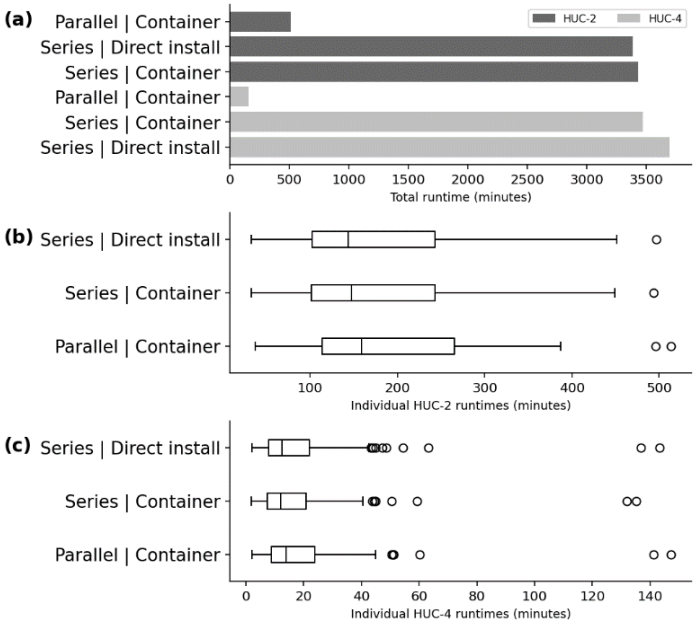

Figure 4: Comparison of successful runtimes for Riparian Land Cover Proportions analyses. (a) Total runtime, (b) distribution of individual HUC-2 runtimes, and (c) distribution of individual HUC-4 runtimes.

3.2 Processing in parallel

Total processing when accessing ArcGIS Python Runtime with an Apptainer SIF image in parallel was 513.42 minutes with HUC-2 sub-tasks and 154.8 minutes with HUC-4 sub-tasks. Individual sub-task processing times ranged from 36.8 to 513.4 minutes with an average of 203.7 minutes (σ = 141.9 minutes) with HUC-2 sub-tasks and from 2.1 to 147.2 minutes with an average of 18.78 minutes (σ = 16.9 minutes) with HUC-4 sub-tasks (Fig. 4). Achieved parallel speedup was 6.67 when processing was partitioned by HUC 2's and 22.55 when split by HUC 4's. While speedup increased with the number of processing units, efficiency fell from 37% (HUC 2 split) to 17.6% (HUC4).

Processing did not complete for eight HUC-2 and 99 HUC-4 sub-tasks when processing in parallel with a direct installation of ArcGIS Server. This was expected as preliminary tests had similar issues when processing with this configuration.

4 Discussion

4.1 Performance

Dividing a large task into manageable sub-tasks is a common strategy when faced with a large analysis or limited computing resources (Lei, 2021;Müller et al., 2013). Often, processing each subtask in series on a single workstation is the only viable option. Total processing time can be significantly reduced by processing sub-tasks in parallel. Our fastest total processing time

https://doi.org/10.5194/egusphere-2025-1558
Preprint. Discussion started: 21 May 2025

was observed when processing at the HUC-4 scale in parallel. When compared to the time to process in-series with a container, we achieved a parallel speedup of 22.55. This scale strikes a balance between advantages of speedup and the effort required to manage and merge sub-tasks. HUC-2 offered less speedup but more efficiency because it requires fewer compute resources (i.e., 18 jobs versus 202 jobs). As the subtask size—in this case spatial scale—decreases and sub-task number increases, so too does the speedup associated with parallelization. Parallel processing times mirrored the longest sub-task processing time. Despite an average HUC4 processing of only 19 minutes, the total runtime of 154 minutes is dictated by HUC 0512 which takes 147 minutes to complete. Variation in stream feature density, feature complexity, and reporting unit size cause these significant differences in sub-task processing times. Load imbalance and increased communication latency account for the decrease in efficiency as sub-task size decreases and quantity increases (i.e. HUC 2 to HUC4) (Tang and Wang, 2020). Dividing our data into even subsets could improve efficiency. However, this approach would be increasingly computationally complex and disrupt the integrity of the reporting unit limiting result interpretability.

Processing in parallel succeeded with the use of containers whereas it consistently failed to complete with a direct install of ArcGIS Server. No error message was produced, but processing routinely stalled for many jobs. The error seemed to interrupt all other currently running jobs on the same node at the completion of any one job on that node. The next job in the queue would start on the newly available resources from the finished job and would run to completion. ArcGIS Server uses various temporary and lock files within its install directory and uses a virtual framebuffer (i.e., Xvfb) to run headless in Linux. Our assumption is that without the isolated environments that containers offer, some combination of resource contention between the temporary files, lock files, or the use of virtual frame buffer introduces a critical conflict and causes processing to halt. The use of containers showed either marginal impact or a slight boost to performance similar to other studies (Le and Paz, 2017;Torrez et al., 2019;Wang et al., 2019).

4.2 Lessons learned and limitations

ATtILA v3 was developed for use in Esri ArcGIS Pro, but considerations were made to ensure compatibility while running on an HPC system. For example, workflows can reference Esri objects like the current map document or current workspace that have default values within ArcGIS Pro. Toolbox authors may rely on these objects having a predefined value that will lead to errors when they are not explicitly defined when running outside of ArcGIS. Additionally, we have found that spatial data projections that may happen automatically (i.e., “on the fly”) in the desktop ArcGIS Pro environment may not when code is executed outside of that environment. Esri recommends all inputs be in the same projection and coordinate system for analysis (Esri, 2025a). Each sub-task requires a unique scratch and output workspace to avoid conflicts when writing intermediate and output files. Concurrent reading of the same input workspace did not pose the same issues. Attention to organization and file structure is useful for managing workflows with a high number of sub-tasks. In our testing, we have found that the issues we encountered in our HPC environment were consistent with using ArcGIS toolboxes outside of ArcGIS Pro on a Windows system.

https://doi.org/10.5194/egusphere-2025-1558
Preprint. Discussion started: 21 May 2025

Overall run time was reduced with smaller spatial scale in parallel; however, the number of output files to manage and merge will obviously increase along with the number of sub-task units. Attention to file structure and naming conventions can alleviate some of this burden.

The setup we have described uses multiple, concurrent tasks where a task is limited by the resources available within a single node. That is, multiple tasks can be executed on one node, but one task cannot use the resources of multiple nodes. While this is a limitation, there are advantages to this workflow as queues for larger, distributed jobs can often leave individual HPC nodes idle. Jobs that request fewer resources generally start faster and can allow for more efficient use of HPC resources. We limited our testing to ArcGIS toolboxes. More efficiencies may be gained by exploring additional Python libraries in conjunction with custom ArcPy code (Zieg and Zawada, 2021).

The RLCP tool requires minimal data from outside the area of interest, so we were confident in our ability to process them in a divide-and-conquer manner using a simple buffer on our input datasets. This is often achievable in GIS particularly when there is a confined unit to subset an analysis with (e.g., county boundaries, watershed boundaries). Analyzing data in subsets is limited when the analysis is dependent on information from outside the unit's boundary without providing sufficient overlap data. For example, network analyses (Armstrong et al., 1992) and landscape connectivity (Koen et al., 2019) are particularly difficult to accurately measure using data subset by arbitrary geographies as patterns well outside, and with unknown distance from, the area of interest can influence the connectivity type. Care must be taken to ensure an appropriate sub-task unit is used for each analysis.

Esri's software is predominately proprietary and commercially licensed. EPA has an enterprise agreement (EA) with Esri that allows for an unlimited number of licenses for certain products over a specified term. We are unsure if this work would have been financially feasible without an EA. The authors confirmed this work falls within the terms and conditions of EPA's EA with Esri; however, other users of the technology discussed in this paper should consider contract and licensing implications.

5 Conclusion

ArcGIS toolboxes are useful packages to develop, share, and document geospatial workflows, but there is little documented use of them in Linux HPC environments, the dominant HPC environment. Esri supports Linux with their Server and Enterprise products; however, concurrent use of their software in an HPC environment is limited. This work presents a method to use ArcGIS toolboxes, or other ArcPy based Python code, in a Linux HPC environment. With the use of container technology, users can initiate multiple jobs in parallel. For workflows that are easily divided into sub-tasks, processing time can be drastically reduced. This solution is not intended to advance efficiency in a particular GIS analysis, but this solution does offer

https://doi.org/10.5194/egusphere-2025-1558
Preprint. Discussion started: 21 May 2025

the possibility of incorporating popular software into HPC workflows and can serve as a starting point for more GIS users to embrace HPC.

Appendices

Table A1. ATtILA parameters used to measure riparian land cover proportions.

Reporting_unit_feature	rf"/{input_basepath}/HUC12_CONUS"
Reporting_unit_ID_field	"HUC_12"
Land_cover_grid	r"/input_basepath/Annual_NLCD_LndCov_2023_CU_C1V0.tif"
Land_cover_classification_scheme	"NLCD LAND"
Land_cover_classification_file	rf"/{attila_basepath}/ATtILA2/ToolboxSource/LandCoverClassifications/NLCD LAND.xml"
Report_metrics_for_these_classes	"'agr - [pagr] Agriculture';'NI - [NINDEX] All Natural Land Use;'for - [pfor] Forest'"
Stream_features	rf"/{input_basepath}/StrmLine" + ';' + r"/{input_basepath}/StrmArea"
Buffer_distance	"45 Meters"
Enforce_reporting_unit_boundaries	"true"
Output_table	rf"/{output_basepath}/HUC12_RLCP"
Processing_cell_size	"30"
Snap_raster	rf"/{input_basepath}/HUC12_CONUS"
Select_options	'"LOGFILE - Record Process Steps Taken During Metric Calculation"'

Code availability

Code for creating our Apptainer SIF is available at https://doi.org/10.5281/zenodo.15066384 (US EPA, 2025) and maintained in a GitHub repository (https://github.com/USEPA/hpc-apptainer-arcgis). The ATtILA toolbox is available at https://doi.org/10.5281/zenodo.8048192 (US EPA, 2023a) and maintained in a GitHub repository (https://github.com/USEPA/ATtILA2). The ATtILA user guide is maintained as a GitHub Wiki (https://github.com/USEPA/ATtILA2/wiki) (US EPA, 2023b). Data developed for this project was for example only and can easily be recreated with the same or different parameters with the ATtILA toolbox.

https://doi.org/10.5194/egusphere-2025-1558
Preprint. Discussion started: 21 May 2025

Author contribution

JB designed the experiment and performed the analysis. JB prepared the manuscript with contributions from all authors. JT and JB developed figures. DE, JT, and JB developed code. SL managed software licensing.

Competing interests

The authors declare that they have no conflict of interest.

Acknowledgements

We acknowledge the support of EPA's High-End Scientific Computing Support Service for their assistance with configuring Atmos and Apptainer and Ed Anderson for his review of the technical specifications of Atmos described in this paper. We acknowledge EPA's National Geospatial Support Team for assistance with software licensing and configuring ArcGIS Server. Two colleagues, James Wickham and David Smith, provided critical reviews and comments to this paper and we acknowledge their contributions.

Disclaimer

This paper has been reviewed in accordance with the U.S. Environmental Protection Agency's peer-review policies and approved for submission. Mention of trade names or commercial products does not constitute endorsement or recommendation for use. Statements in this publication reflect the authors' personal views and opinions and should not be construed to represent any determination of policy of the U.S. Environmental Protection Agency.

Financial Support

References

1. Abraham, S., Paul, A. K., Khan, R. I. S., and Butt, A. R.: On the use of containers in high performance computing environments, 2020 IEEE 13th International Conference on Cloud Computing (CLOUD), 284-293, https://doi.org/10.1109/CLOUD49709.2020.00048, 2020.
2. Ahmari, H., Pebworth, M., Baharvand, S., Kandel, S., and Yu, X.: Development of an ArcGIS-Pro Toolkit for Assessing the Effects of Bridge Construction on Overland Soil Erosion, Land, 11, 1586, https://doi.org/10.3390/land11091586 2022.
3. Armstrong, M. P., Densham, P. J., and Ding, Y.: Parallel processing for network analysis: decomposing shortest path algorithms on MIMD computers, Proceedings 5th International Symposium on Spatial Data Handling, 1992, 682-691,

https://doi.org/10.5194/egusphere-2025-1558
Preprint. Discussion started: 21 May 2025

4. Barrios-Hernandez, C. J., Rivas, R., Besseron, X., Diaz, G., Georgiou, Y., and Velho, P.: High-Performance Computing Training by Skills for Advanced Computing Ecosystems, Revista UIS Ingenierías, 23, https://doi.org/10.18273/revuin.v23n1-2024014, 2024.
5. Clematis, A., Mineter, M., and Marciano, R.: High performance computing with geographical data, Parallel Computing, 29, 1275-1280, https://doi.org/10.1016/j.parco.2003.07.001, 2003.
6. Coetzee, S., Ivánová, I., Mitasova, H., and Brovelli, M. A.: Open Geospatial Software and Data: A Review of the Current State and A Perspective into the Future, ISPRS International Journal of Geo-Information, 9, 90, https://doi.org/10.3390/ijgi9020090, 2020.
7. Eager, D. L., Zahorjan, J., and Lazowska, E. D.: Speedup Versus Efficiency in Parallel Systems, Ieee Transactions on Computers, 38, 408-423, https://doi.org/10.1109/12.21127, 1989.
8. Ebert, D., Wade, T.: Analytical Tools Interface for Landscape Assessments (ATtILA) User Manual: https://www.epa.gov/sites/default/files/2015-06/documents/user_guide.pdf, access: 20250320, 2004.
9. Erickson, R. A., Fienen, M. N., McCalla, S. G., Weiser, E. L., Bower, M. L., Knudson, J. M., and Thain, G.: Wrangling distributed computing for high-throughput environmental science: An introduction to HTCondor, PLoS Comput Biol, 14, e1006468, https://doi.org/10.1371/journal.pcbi.1006468, 2018.
10. Esri: Independent Report Highlights Esri as Leader in Global GIS Market: https://www.esri.com/about/newsroom/announcements/independent-report-highlights-esri-as-leader-in-global-gis-market/, access: 20250320, 2015.
11. Esri: Coordinate systems, map projections, and transformations: https://pro.arcgis.com/en/pro-app/latest/help/mapping/properties/coordinate-systems-and-projections.htm, access: 20250225, 2025a.
12. Esri. What version of Python is used in ArcGIS?: https://support.esri.com/en-us/knowledge-base/faq-what-version-of-python-is-used-in-arcgis-000013224, access: 20250225, 2025b.
13. Esri: Comparing custom and Python toolboxes: https://pro.arcgis.com/en/pro-app/latest/arcpy/geoprocessing_and_python/comparing-custom-and-python-toolboxes.htm, access: 20250225, 2025c.
14. Fienen, M. N., and Hunt, R. J.: High-throughput computing versus high-performance computing for groundwater applications, Ground Water, 53, 180-184, https://doi.org/10.1111/gwat.12320, 2015.
15. Fleming, J., Marvel, S. W., Supak, S., Motsinger-Reif, A. A., and Reif, D. M.: ToxPi*GIS Toolkit: creating, viewing, and sharing integrative visualizations for geospatial data using ArcGIS, J Expo Sci Environ Epidemiol, 32, 900-907, https://doi.org/10.1038/s41370-022-00433-w, 2022.
16. GeriMiller: ArcGIS and High Performance Computing (HPC): https://community.esri.com/t5/education-blog/arcgis-and-high-performance-computing-hpc/ba-p/883968, access: 20250320, 2023.
17. Gong, J. Y., and Xie, J.: Extraction of drainage networks from large terrain datasets using high throughput computing, Computers & Geosciences, 35, 337-346, https://doi.org/10.1016/j.cageo.2008.09.002, 2009.
18. Hashtarkhani, S., Schwartz, D. L., and Shaban-Nejad, A.: Enhancing Health Care Accessibility and Equity Through a Geoprocessing Toolbox for Spatial Accessibility Analysis: Development and Case Study, JMIR Form Res, 8, e51727, https://doi.org/10.2196/51727, 2024.
19. Healey, R., Dowers, S., Gittings, B., and Mineter, M.: Parallel Processing Algorithms for GIS, https://doi.org/10.1201/9781003062684, 2020.
20. Huerta, E. A., Haas, R., Jha, S., Neubauer, M., and Katz, D. S.: Supporting High-Performance and High-Throughput Computing for Experimental Science, Computing and Software for Big Science, 3, 1-15, https://doi.org/10.1007/s41781-019-0022-7, 2019.
21. Koen, E. L., Ellington, E. H., and Bowman, J.: Mapping landscape connectivity for large spatial extents, Landscape Ecology, 34, 2421-2433, https://doi.org/10.1007/s10980-019-00897-6, 2019.
22. Kurtzer, G. M., Sochat, V., and Bauer, M. W.: Singularity: Scientific containers for mobility of compute, PLoS One, 12, e0177459, https://doi.org/10.1371/journal.pone.0177459, 2017.
23. Le, E., and Paz, D.: Performance analysis of applications using singularity container on sdsc comet, Practice and Experience in Advanced Research Computing 2017: Sustainability, Success and Impact, 1-4, https://doi.org/10.1145/3093338.3106737, 2017.

https://doi.org/10.5194/egusphere-2025-1558
Preprint. Discussion started: 21 May 2025

24. Lei, T. L.: Large scale geospatial data conflation: A feature matching framework based on optimization and divide-and-conquer, Comput Environ Urban, 87, 101618, https://doi.org/10.1016/j.compenvurbsys.2021.101618, 2021.
25. Leonard, P. B., Baldwin, R. F., Homyack, J. A., and Wigley, T. B.: Remote detection of small wetlands in the Atlantic coastal plain of North America: Local relief models, ground validation, and high-throughput computing, Forest Ecology and Management, 284, 107-115, https://doi.org/10.1016/j.foreco.2012.07.034, 2012.
26. Leonard, P. B., Baldwin, R. F., Duffy, E. B., Lipscomb, D. J., and Rose, A. M.: High-throughput computing provides substantial time savings for landscape and conservation planning, Landscape and Urban Planning, 125, 156-165, https://doi.org/10.1016/j.landurbplan.2014.02.016, 2014.
27. Lin, C., Nadi, S., and Khazaei, H.: A large-scale data set and an empirical study of docker images hosted on docker hub, 2020 IEEE International Conference on Software Maintenance and Evolution (ICSME), 371-381, https://doi.org/10.1109/ICSME46990.2020.00043, 2020.
28. Loraamm, R., Downs, J., Anderson, J., and Lamb, D. S.: PySTPrism: Tools for voxel-based space-time prisms, Softwarex, 12, 100499, https://doi.org/10.1016/j.softx.2020.100499, 2020.
29. McKay, L., Bondelid, T., Dewald, T., Johnston, J., Moore, R., and Rea, A.: NHDPlus Version 2: User Guide: https://www.epa.gov/waterdata/nhdplus-national-hydrography-dataset-plus, access: 20250320, 2012.
30. Müller, M., Bernard, L., and Kadner, D.: Moving code – Sharing geoprocessing logic on the Web, ISPRS Journal of Photogrammetry and Remote Sensing, 83, 193-203, https://doi.org/10.1016/j.isprsjprs.2013.02.011, 2013.
31. Openshaw, S., and Turton, I.: High Performance Computing and the Art of Parallel Programming, https://doi.org/10.4324/9780203981436, 2005.
32. Pijanowski, B. C., Tayyebi, A., Doucette, J., Pekin, B. K., Braun, D., and Plourde, J.: A big data urban growth simulation at a national scale: Configuring the GIS and neural network based Land Transformation Model to run in a High Performance Computing (HPC) environment, Environmental Modelling & Software, 51, 250-268, https://doi.org/10.1016/j.envsoft.2013.09.015, 2014.
33. Raad, M.: ArcGIS For Server On Docker [code]: https://github.com/mraad/docker-arcgis, access: 20250225, 2016.
34. Raj, R. K., Romanowski, C. J., Impagliazzo, J., Aly, S. G., Becker, B. A., Chen, J., Ghafoor, S., Giacaman, N., Gordon, S. I., and Izu, C.: High performance computing education: Current challenges and future directions, Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education, 51-74, https://doi.org/10.1145/3437800.3439203, 2020.
35. Saltus, C. L., Reif, M. K., and Johansen, R. A.: Waterquality for ArcGIS Pro Toolbox, https://doi.org/10.21079/11681/45362, 2022.
36. Scheider, S., Ballatore, A., and Lemmens, R.: Finding and sharing GIS methods based on the questions they answer, International Journal of Digital Earth, 12, 594-613, https://doi.org/10.1080/17538947.2018.1470688, 2019.
37. Tang, J. Y., and Matyas, C. J.: Arc4nix: A cross-platform geospatial analytical library for cluster and cloud computing, Computers & Geosciences, 111, 159-166, https://doi.org/10.1016/j.cageo.2017.11.011, 2018.
38. Tang, W., and Wang, S.: Navigating High Performance Computing for Geospatial Applications, High Performance Computing for Geospatial Applications, 1-5, https://doi.org/10.1007/978-3-030-47998-5_1, 2020.
39. Top 500: List Statistics - Operating system Family: https://top500.org/statistics/list/, access: 20250320, 2024.
40. Torrez, A., Randles, T., and Priedhorsky, R.: HPC container runtimes have minimal or no performance impact, 2019 IEEE/ACM International Workshop on Containers and New Orchestration Paradigms for Isolated Environments in HPC (CANOPIE-HPC), 37-42, https://doi.org/10.1109/CANOPIE-HPC49598.2019.00010, 2019.
41. Trisovic, A., Lau, M. K., Pasquier, T., and Crosas, M.: A large-scale study on research code quality and execution, Sci Data, 9, 60, https://doi.org/10.1038/s41597-022-01143-6, 2022.
42. US EPA: ATtILA2: ATtILA for ArcGIS Pro (v3.0.0) [code], https://doi.org/10.5281/zenodo.15079386, 2023a.
43. US EPA: Analytical Tools Interface for Landscape Assessments (ATtILA) User Guide: https://github.com/USEPA/ATtILA2/wiki, access: 20250320, 2023b.
44. US EPA: hpc-apptainer-arcgis [code], https://doi.org/10.5281/zenodo.15066385, 2025.
45. US Geological Survey: Watershed Boundary Dataset [data set]: https://www.usgs.gov/national-hydrography/watershed-boundary-dataset, access: 20250204.
46. US Geological Survey: Annual National Land Cover Database (NLCD) Collection 1 Products [data set], https://doi.org/10.5066/P94UXNTS, 2024.

https://doi.org/10.5194/egusphere-2025-1558
Preprint. Discussion started: 21 May 2025

47. Vandewalle, R. C., Barley, W. C., Padmanabhan, A., Katz, D. S., and Wang, S.: Understanding the multifaceted geospatial software ecosystem: a survey approach, International Journal of Geographical Information Science, 35, 2168-2186, https://doi.org/10.1080/13658816.2020.1831514, 2020.
48. Wang, D., Berry, M., Buchanan, N., and Gross, L.: A GIS-enabled distributed simulation framework for high performance ecosystem modeling, ESRI International User Conference, 2006, 1-5,
49. Wang, D., Zhao, Z., Shaw, S.-L., Ragghianti, G., and Wei, Y.: Building a high performance ArcGIS cluster and its application to climate data processing, Proceedings of the GeoComputation 2013 Conference, Wuhan, China, 2013, 23-25,
50. Wang, Y., Evans, R. T., and Huang, L.: Performant container support for HPC applications, Practice and Experience in Advanced Research Computing 2019: Rise of the Machines (Learning), 1-6, https://doi.org/10.1145/3332186.3332226, 2019.
51. Yoo, A. B., Jette, M. A., and Grondona, M.: Slurm: Simple linux utility for resource management, Workshop on job scheduling strategies for parallel processing, 44-60, https://doi.org/10.1007/10968987_3, 2003.
52. Zheng, M. R., Tang, W. W., Lan, Y., Zhao, X., Jia, M. J., Allan, C., and Trettin, C.: Parallel Generation of Very High Resolution Digital Elevation Models: High-Performance Computing for Big Spatial Data Analysis, Big Data in Engineering Applications, 44, 21-39, https://doi.org/10.1007/978-981-10-8476-8_2, 2018.
53. Zhu, A. X., Zhao, F. H., Liang, P., and Qin, C. Z.: Next generation of GIS: must be easy, Annals of Gis, 27, 71-86, https://doi.org/10.1080/19475683.2020.1766563, 2021.
54. Zieg, J., and Zawada, D. G.: Improving ESRI ArcGIS Performance of Coastal and Seafloor Analyses with the Python Multiprocessing Module, Journal of Coastal Research, 37, 1288-1293, https://doi.org/10.2112/Jcoastres-D-21-00026.1, 2021.
55. Zurqani, H. A., Post, C. J., Mikhailova, E. A., Cope, M. P., Allen, J. S., and Lytle, B. A.: Evaluating the integrity of forested riparian buffers over a large area using LiDAR data and Google Earth Engine, Sci Rep, 10, 14096, https://doi.org/10.1038/s41598-020-69743-z, 2020.