

Original Software Publication

CyberGIS-Compute: Middleware for democratizing scalable geocomputation

Alexander C. Michels^a, Anand Padmanabhan^a, Zimo Xiao^{a,b}, Mit Kotak^{a,c}, Furqan Baig^a, Shaowen Wang^{a,*}

^a CyberGIS Center for Advanced Digital and Spatial Studies, University of Illinois Urbana-Champaign, 1301 W Green St, Urbana, IL, 61801, USA

^b Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA, 15213, USA

^c Center for Computational Science and Engineering, Massachusetts Institute of Technology, Cambridge, MA, 02139, USA

ARTICLE INFO

Keywords:

CyberGIS
High-performance computing
Middleware
Scientific workflow

ABSTRACT

CyberGIS—geographic information science and systems (GIS) based on advanced cyberinfrastructure—is becoming increasingly important to tackling a variety of socio-environmental problems like climate change, disaster management, and water security. While recent advances in high-performance computing (HPC) have the potential to help address these problems, the technical knowledge required to use HPC has posed challenges to many domain experts. In this paper, we present CyberGIS-Compute: a geospatial middleware tool designed to democratize HPC access for solving diverse socio-environmental problems. CyberGIS-Compute does this by providing a simple user interface in Jupyter, streamlining the process of integrating domain-specific models with HPC, and establishing a suite of APIs friendly to domain experts.

1. Motivation and significance

Work across many disciplines is increasingly dependent on geographic information science and systems (GIS) based on advanced cyberinfrastructure (CI), or cyberGIS [1]. While cyberGIS enables computationally intensive problem solving through geospatial big data, large-scale analysis and modeling, and artificial intelligence (AI) approaches, many are unable to benefit from cutting-edge advances in cyberGIS due to the high technical barriers that advanced CI imposes. Domain experts utilizing GIS come from a wide range of disciplines including for example agriculture, hydrology, and public health and they cannot all be expected to have the technical expertise necessary to interact directly with advanced CI. However, what they may lack in technical skills, they more than make up for in domain expertise, so empowering their workflows with advanced CI can yield tremendous benefits to the diverse scientific community.

While there are a variety of tools designed for interacting with High-Performance Computing (HPC) resources, none of them exactly meets the needs of domain experts in geospatial science and related fields. Tools like Docker [2] and Apptainer [3] address reproducibly interacting with HPC through the command line, but these present an insurmountable technical barrier for a significant number of domain experts.

Tools like Apache Airflow [4] allow users to seamlessly run workflows, but academic HPC resources do not allow general users to run code that has not been vetted. We need a solution that provides reproducible access to HPC resources, has a low technical barrier, and is configurable but not overly open.

Providing reproducible execution environments for scientific workflows has been a priority in the scientific community for years to help address the reproducibility crisis [5]. The geospatial community has developed a number of science gateways to combat the crisis [6–10], including our own CyberGIS-Jupyter approach [8,11–13], but these capabilities are often not well-equipped to support reproducibility in large-scale computational analysis. HPC resources are made available to researchers and scientific communities through advanced CI such as ACCESS [14], but applying for and utilizing these resources comes with significant technical barriers. Our goal is to provide an accessible, reproducible bridge between science gateways and HPC.

CyberGIS-Compute [15,16] is a generic middleware tool designed to democratize access to scalable geospatial computing and HPC resources for diverse domain experts. We have seen success by providing a simple user interface built on Jupyter notebooks to reduce UI development [17], streamlining the process of integrating domain-specific models with HPC, and establishing a suite of APIs friendly to domain experts,

* Corresponding author.

E-mail address: shaowen@illinois.edu (S. Wang).

<https://doi.org/10.1016/j.softx.2024.101691>

Received 4 November 2023; Received in revised form 11 March 2024; Accepted 12 March 2024

Available online 28 March 2024

2352-7110/© 2024 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY license (<http://creativecommons.org/licenses/by/4.0/>).

which has enabled computation- and data-intensive work in a variety of domains such as remote sensing of environment [17], spatial accessibility [17], and hydrology.

CyberGIS-Compute has two complementary uses: (1) providing a simple interface for model developers and end-users to interact with CI and (2) providing a framework for sharing computationally intensive executable workflows. Domain experts are able to convert their workflows into CyberGIS-Compute models without needing expertise in CI by simply adding a JSON manifest to their Github repository giving meta-data about the code and how it can be run. End-users are then able to execute and reproduce the workflows using our Jupyter-based graphical user interface (GUI). This gives users the familiar Jupyter interface that is used throughout the geosciences [6–10] with the ability to connect to any HPC system that uses SLURM, allowing them to harness the computational power of their local HPC resources or leading-edge HPC resources provided by ACCESS [14] like ACES [18].

2. Software description

CyberGIS-Compute is a middleware tool that allows users to execute models on HPC resources using Apptainer (formerly Singularity) [3] and SLURM [19] from a GUI within Jupyter notebooks [17]. CyberGIS-Compute is made up of two fundamental components: the Core which is a server that executes and manages jobs submitted by users and the software development kit (SDK) which provides a user interface. The Core server, discussed further in Section 2.1, is a REpresentational State Transfer (REST) Application Programming Interface (API) server written in Typescript with Express [20]. The SDK, discussed further in Section 2.2, is written in Python 3 and available on the Python Package Index (PyPi).

2.1. CyberGIS-Compute Core

CyberGIS-Compute Core abstracts away the technical complexities of HPC resources by providing Apptainer based job automation [3]. Its architecture consists of three layers: Application Programming Interface (API) Server, Maintainer Pool, and Connection Pool, which work together sequentially to submit user applications to advanced CI resources. This architecture is illustrated in Fig. 1.

The API Server exposes a set of standardized RESTful APIs for basic HPC operations such as job submission and data upload. To access the API, users must have credentials for a JupyterHub [17] within our allowlist and authentication is performed using JupyterHub's tokens and API, all of which is handled for users by the SDK transparently. Job operation requests received by the API are pushed into the Job Queue and consumed by a Maintainer Worker process in the Maintainer Pool. The Maintainer Pool creates and manages multiple Maintainer Worker processes that contain the automated logic behind each abstracted HPC operation, including fetching contributed models from GitHub, setting up HPC jobs, transferring data using Globus [21], submitting jobs, and monitoring job status. Maintainers in our Maintainer Pool run asynchronously and can support dozens of jobs concurrently. Future work will add another layer to support more complex load balancing, fairness, and throttling mechanisms to the Maintainer and Connector pools. Each Maintainer Worker process communicates with HPC resources using a shared abstract SSH Connector process. Because a Connector process is shared between many different Maintainer Workers, the SSH connection is prone to race conditions and file contamination. In order to solve this problem, a Connector Pool is implemented to communicate with HPC resources, provide mutex locks, and file-level isolation.

CyberGIS-Compute leverages SLURM Workload Manager [19] and Apptainer containerization [3] to simplify and standardize executing models on advanced CI resources. This provides models developed for CyberGIS-Compute a consistent environment across HPC resources. We expose a standardized set of APIs for users to define a job's computing resources and virtualization environment. From the users' inputs, the Maintainer Worker is able to translate users' requests into SLURM scripts that request computing resources from HPC and sets up an Apptainer containerized environment from a library of images with built-in geospatial programming environments. Docker is not generally supported on HPC due to security issues [3], but existing Docker images can be easily converted to Apptainer images for usage in CyberGIS-Compute and we can add support for Docker if HPC support for Docker is available. A computing resource quota system is implemented to prevent potential exploitation of the resource.

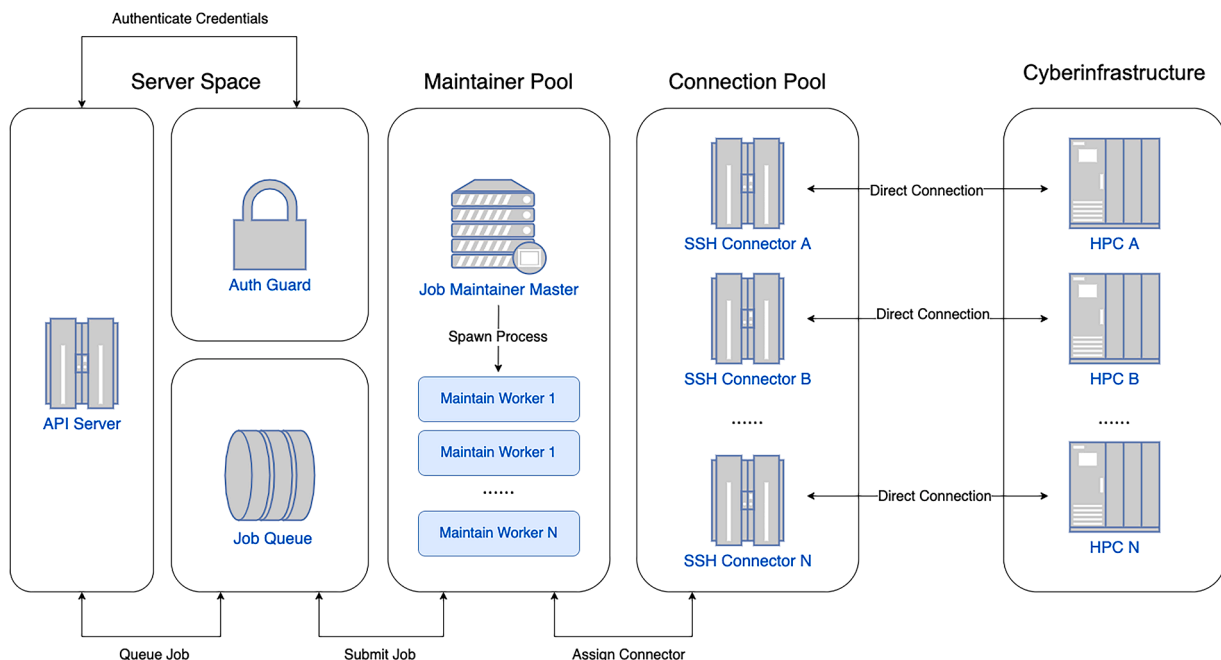


Fig. 1. Architecture of the CyberGIS-Compute Core.

2.2. CyberGIS-Compute SDK

The CyberGIS-Compute SDK is a client-side tool for utilizing CyberGIS-Compute. The SDK communicates with the Core using Python 3 `http.client` module in the standard library. A connection is established to the Core server by instantiating a `CyberGISCompute` object as shown in Cell 1 of Fig. 2. The Core server checks the JupyterHub against an allowlist and then authenticates the user's token against the JupyterHub to verify the request. Once instantiated, the `CyberGISCompute` object offers a variety of functionalities to list available models, download results and more, while most interactions take place through the GUI.

The GUI of CyberGIS-Compute, shown in Fig. 2, allows users to

submit and monitor jobs. Job submission steps that would otherwise require the use of a command line have been simplified into different aspects of our GUI. For example, the “Job Configuration” tab is used to specify information about a job before it is submitted. This can include the computing resource to use, SLURM [19] configurations, and data to upload. Once configured, jobs can be submitted by clicking the “Submit Job” button as seen in Fig. 2. Live updates on jobs are provided through the “Your Job Status” tab, letting each user know when their job goes through each step of the CyberGIS-Compute architecture: the job queue, registered with the computing resource, started, and finished. The tab also notifies the user when the job has completed and displays an excerpt from the job's logs.

After a job completes, users are able to fetch the results using the

```
[1]: from cybergis_compute_client import CyberGISCompute

cybergis = CyberGISCompute(url="cgjobsup.cigi.illinois.edu", isJupyter=True,
                             protocol="HTTPS", port=443, suffix="v2")

[2]: cybergis.show_ui()
```

Job Configuration | Your Job Status | Download Job Result | Your Jobs | Past Results

Welcome to CyberGIS-Compute

A scalable middleware framework for enabling high-performance and data-intensive geospatial research and education on CyberGIS-Jupyter

Job Templates: covid-access

covid-access Job Description: Calculates travel-time from hospitals and then calculates spatial accessibility to hospital beds for the entire state of Illinois. We calculate travel-time and aggregate spatial accessibility in parallel using 4 CPUs and about 64-80GB of memory. Read about the paper here: <https://doi.org/10.1186/s12942-020-00229-x>

expanse_community HPC Description: none

Estimated Runtime: ~20 minutes

Computing Resource

Computing Resource: expanse_community

Slurm Computing Configurations

All configs are optional. Please refer to Slurm official documentation at <https://slurm.schedmd.com/sbatch.html>

time (Minutes) 120

cpu_per_task 4

memory_per_cpu (GB) 20

☐ receive email on job status? example@illinois.edu

Submit Job

Fig. 2. A screenshot of the CyberGIS-Compute SDK's interface in a Jupyter notebook [17].

“Download Job Results” tab of the GUI. Users can download all of the contents of the job result folder or just sub-folders like the full SLURM logs. Once a user clicks the “Download” button, CyberGIS-Compute initiates a Globus [21] file transfer from a HPC resource to the user’s file system. When the download completes, the path to the downloaded data is displayed on the GUI and provided to the user through the ‘recentDownloadPath’ attribute of the CyberGIS-Compute object. This allows users to develop notebooks that submit computationally intensive jobs through CyberGIS-Compute and then perform further analysis on the results of the job locally.

The GUI has two tabs dedicated to interacting with jobs that have previously been executed. The “Your Jobs” tab keeps a record of each user’s previously submitted jobs, allowing users to review and download older information. Jobs are listed along with metadata describing the underlying model, parameters, and other information about the model. Each job has a “Restore” button that allows users to populate the GUI with that job’s information and re-download the results. The “Past Results” tab lists all of the previous result folders available with a “Download” button for convenience. These functionalities were developed so that users could access the configuration and results of jobs in the event of a browser refresh or kernel disconnection and is particularly useful for long-running jobs because users can now close their browsers and come back to check the results.

3. Illustrative example

In this section we illustrate how a model is contributed and used in CyberGIS-Compute. At the beginning of the COVID-19 pandemic, Kang et al. 2020 [16] implemented a model for calculating spatial accessibility to hospital ventilators in the state of Illinois, USA. Spatial accessibility is an interdisciplinary field of research that explores how access to resources and services varies across space. This research is conducted across fields including economics, geography, and public health to study access to health care [22,23], food [24], electric vehicle charging stations [25], and more. Spatial accessibility analyses depend on mobility and travel-time information that is computationally intensive to compute which has forced many researchers to limit their analysis to small geographic areas such as a city or several counties. Being an interdisciplinary field facing challenges due to limited computational resources makes spatial accessibility an excellent candidate for CyberGIS-Compute. Kang et al. wanted to share their workflow on CyberGISX science gateway to ensure reproducibility, but because the computational intensity of the workflow was too great for the science gateway, were forced to make a simplified version of the model which only considers the city of Chicago, IL, USA.

Using the Jupyter notebook shared on Github, we were able to create a CyberGIS-Compute model for the workflow. First, we converted the Jupyter notebook to a Python script using the Jupyter nbconvert tool [17]. Second, we needed to make minor changes to the code so that the outputs of the script were put into a results folder so that the data can be retrieved. Third, we needed to specify a “manifest.json” file which provides metadata and describes how to execute the model with CyberGIS-Compute.

The manifest.json for the COVID-19 spatial accessibility model is shown in Listing 1. The first three entries (name, description, estimated runtime) are to inform end-users about the model. The middle section of the manifest specifies how the model will be run. The “container” describes the name of an Apptainer image, from a curated list of images, in which the analysis will execute. The “execution_stage” is one of three stages that can be specified in a CyberGIS-Compute model (pre-processing, execution, and post-processing) and describes commands to be executed in each stage. In this case, the only stage is the execution of the Python script “main.py”. Next, “slurm_input_rules” describe options which are provided to end-users to specify the maximum amount of time, CPUs, and memory that the model will be allocated. The “require_upload_data” tells CyberGIS-Compute that users do not need to

upload any data to perform the analysis. Lastly, the “supported_hpc” gives a list of HPC resources on which the model can be executed (ROGER [26] at the University of Illinois Urbana-Champaign, Expanse [27], and Anvil [28]) while the “default_hpc” tells CyberGIS-Compute to use Expanse if no other HPC is given.

Once the model is entered into the database for a Core server, it is available for end-users to run. Each model is entered into the Core server’s database with a short name, in this case “covid-access”. When a user connects to our CyberGIS-Compute server, launches the GUI, and selects the “covid-access” model from the “Job Template” dropdown, the GUI is populated with information from the manifest as shown in Fig. 1. The description of the model and estimated runtime are filled in and the appropriate sliders are made available under the “Slurm Computing Configurations” section. However, an end-user is able to rely on the default values provided by the model creator and simply click “Submit Job” after the model is selected in the dropdown.

```

1 {
2   "name": "COVID-19 spatial accessibility",
3   "description": "Calculates travel-time from hospitals and then calculates spatial
4     accessibility to hospital beds for the entire state of Illinois. We calculate travel-time and
5     aggregate spatial accessibility in parallel using 4 CPUs and about 64-80GB of memory.
6     Read about the paper here: https://doi.org/10.1186/s12942-020-00,229-x",
7   "estimated_runtime": "~20 min",
8   "container": "cybergisx-0.4",
9   "execution_stage": "MPLBACKEND=Agg python main.py",
10  "slurm_input_rules": {
11    "time": {
12      "max": 180,
13      "min": 60,
14      "default_value": 120,
15      "step": 1,
16      "unit": "Minutes"
17    },
18    "cpu_per_task": {
19      "max": 4,
20      "min": 4,
21      "default_value": 4,
22      "step": 1
23    },
24    "memory_per_cpu": {
25      "max": 20,
26      "min": 16,
27      "default_value": 20,
28      "step": 1,
29      "unit": "GB"
30    }
31  },
32  "require_upload_data": false,
33  "supported_hpc": ["roger_community", "expanse_community",
34    "anvil_community"],
35  "default_hpc": "expanse_community"
36 }

```

Listing 1. The manifest for the COVID-19 Spatial Accessibility model that implements the workflow from Kang et al. 2022 [29].

Once a job is submitted, the GUI changes tabs to the “Your Job Status” tab which provides live updates on the running job as seen in Fig. 3. The information just below “Your Job is Here!” provides information on the parameters sent to the Core server including the SLURM parameters from the GUI’s sliders. The “job events” section of the page tracks the job as it reaches the Core server’s queue, the model’s code is sent to the HPC, executes on the HPC, and ends. After the job ends, the “job logs” portion of the page displays a short excerpt of the SLURM logs. To obtain the full logs and any job outputs, the “Download Job Result” tab provides a dropdown allowing users to choose between downloading the whole folder or sub-directories and a “Download” button which initiates a Globus transfer bringing the results from the HPC to the Jupyter environment where the CyberGIS-Compute SDK is being executed.

While running on the HPC center, the COVID-19 Spatial Accessibility model calculates driving-time catchments around each hospital in the

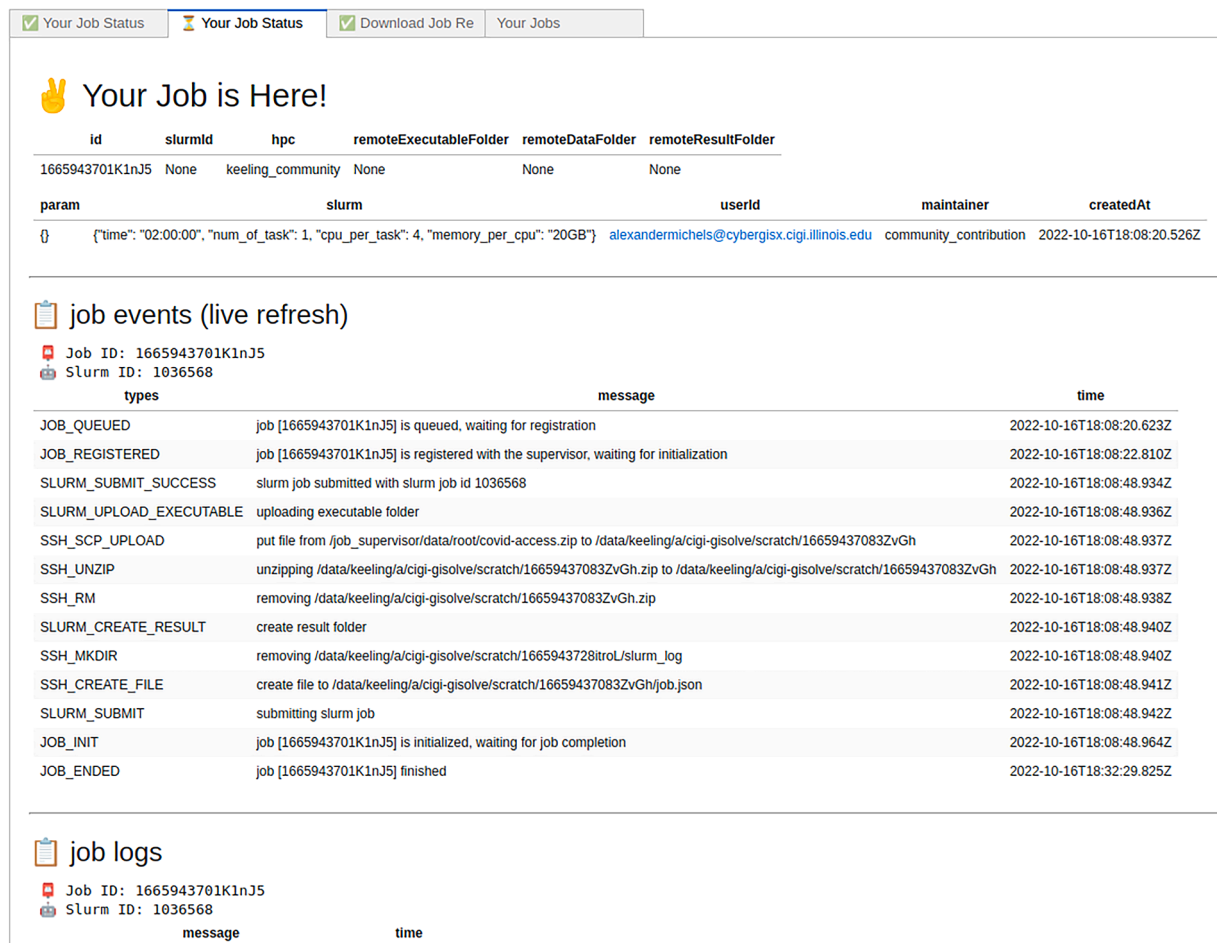


Fig. 3. The “Your Job Status” tab of the CyberGIS-Compute Graphical User Interface (GUI) after the “covid-access” (COVID-19 Spatial Accessibility) job has been submitted and finished running.

dataset using OpenStreetMap [30] data and the Python OSMnx package [31]. These calculations are performed in parallel thanks to the Python multiprocessing library, a part of the standard library. Then, the model uses the travel-time catchments to calculate spatial accessibility using the Enhanced Two-Step Floating Catchment Area (E2SFCA) [32] method. This results in a metric for each spatial unit (in this case a hexagonal grid [29]) which can be interpreted as a ratio of hospital ventilators to people. The pattern of spatial accessibility to hospital ventilators calculated by the model is mapped in Fig. 4. The results from this analysis can be downloaded by users using the “Download Job Results” tab of the GUI. Additional use-cases of CyberGIS-Compute can be viewed on our Model Gallery or the CyberGIS’s CyberGIS-Compute Showcase community page.

4. Impact

CyberGIS-Compute has enabled domain experts to create and share complex computational workflows. Lyu et al. 2022 utilized CyberGIS-Compute to run a hybrid deep learning model on remote sensing data using GPUs Bridges-2 HPC resource and retrieve results, all from a notebook based on CyberGIS-Jupyter [33]. Lyu et al. noted that using CyberGIS-Compute “makes the data fusion workflow scalable and reproducible, and accessible by domain experts who may have limited programming background and cyberinfrastructure knowledge” [33]. In addition to enabling new research questions and directions, CyberGIS-Compute has been used to make an already published computationally intensive workflow available [29] and hydrological models have been developed as a collaboration with HydroShare [6].

Beyond this work, there are a number ongoing projects utilizing CyberGIS-Compute and we believe that the lowered technical barriers will enable domain experts to analyze and model geospatial phenomena at unprecedented spatio-temporal scales.

CyberGIS-Compute has been installed and used on the CyberGIS-Jupyter for Water [34] and I-GUIDE Platform [35] science gateways. These gateways support research and education in the geosciences and beyond. Beyond these two gateways, we have been working to promote our software and reach a broader audience. We have given a talk on the work at the Gateways conference [12], hosted a demo at the SIGSPATIAL conference [16], gave two invited talks for the I-GUIDE Virtual Consulting Office series [36,37], ran a workshop at the 2023 Annual Meeting of the American Association of Geographers (AAG) [38], and conducted a tutorial at the Accelerating Computing for Emerging Sciences (ACES) 2023 Workshop [39]. These various workshops and talks have yielded multiple collaborations, growing our list of supported models. Both the Core and SDK are available on Github and the SDK is available to download on PyPi, so anyone with a JupyterHub is able to deploy their own Core server, with their own set of models, and connect it to any HPC resource that supports SLURM.

Our work has also inspired a similar project, CyberGIS-Cloud for managing workflows on commercial cloud [40]. Whereas CyberGIS-Compute is focused on providing a simple interface for executing geospatial workflows on HPC resources, CyberGIS-Cloud is working to provide users with access to cloud storage and clusters using Hadoop [41], Spark [42], and Dask [43]. Together, these projects continue to enable researchers and educators to harness both HPC and cloud resources.

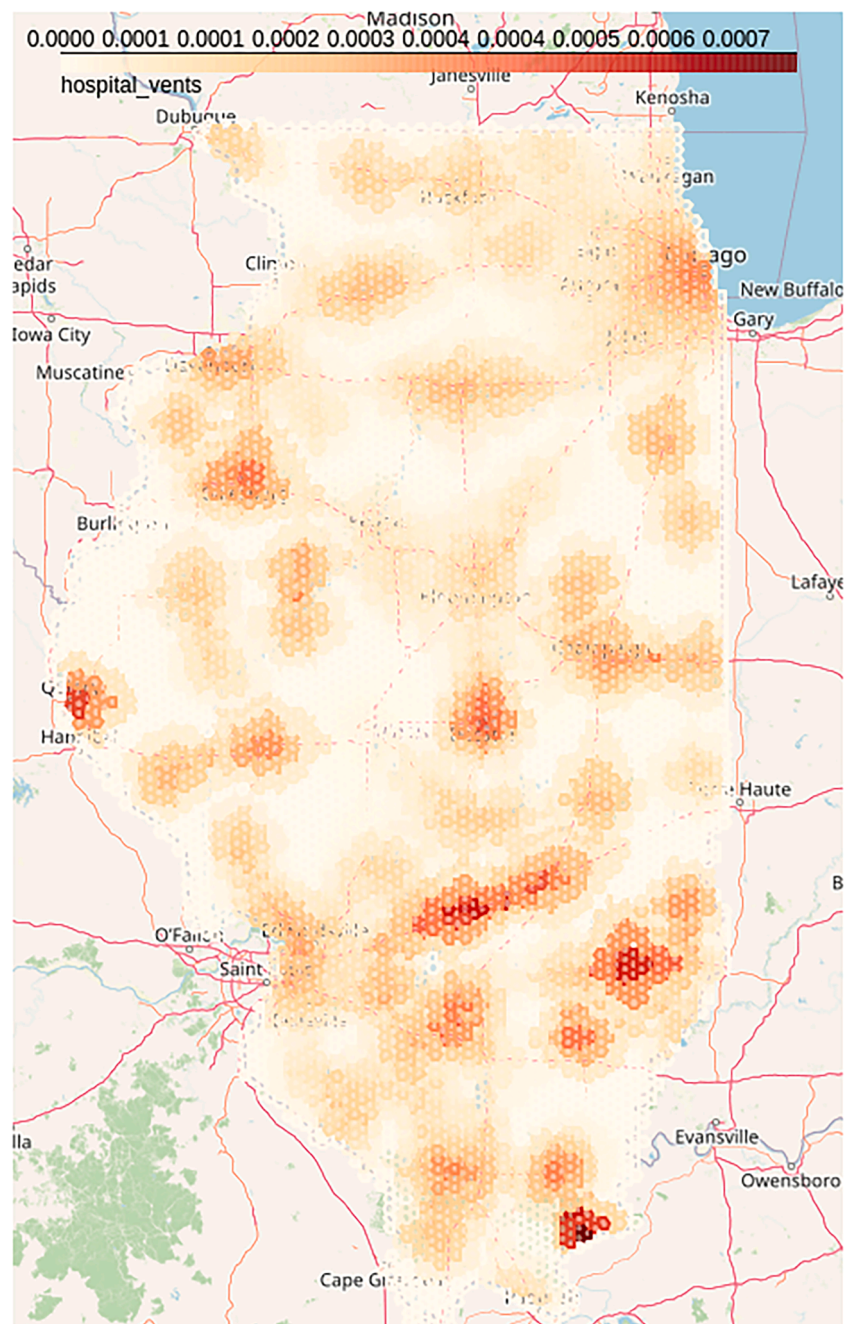


Fig. 4. The map of spatial accessibility to hospital ventilators calculated by the COVID-19 Spatial Accessibility model using CyberGIS-Compute.

5. Conclusions

In this paper, we describe CyberGIS-Compute, a middleware toolkit for democratizing scalable geocomputation. The software comprises two key components, a Typescript server which executes code on HPC resources and an SDK with graphical user interface written in Python. CyberGIS-Compute lowers the barriers to utilizing HPC by providing a streamlined model contribution mechanism and a simplified graphical user interface for executing computation jobs while providing a framework for sharing reproducible scientific workflows. The software has a growing user-base and library of models and has already inspired similar work aimed at democratizing access to advanced CI.

CRediT authorship contribution statement

Alexander C. Michels: Writing – review & editing, Writing – original draft, Validation, Software, Methodology, Investigation, Conceptualization. **Anand Padmanabhan:** Writing – review & editing, Validation, Supervision, Project administration, Methodology, Investigation, Conceptualization. **Zimo Xiao:** Software, Methodology, Investigation, Conceptualization. **Mit Kotak:** Software, Methodology, Investigation, Conceptualization. **Furqan Baig:** Validation, Software, Methodology, Investigation. **Shaowen Wang:** Writing – review & editing, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgements

This research is based in part on work supported by the U.S. National Science Foundation (NSF) under grant numbers 2118329, 2112356, and 2321070. The authors would like to thank the dedicated personnel at several HPC centers and NSF ACCESS who support the HPC resources utilized by CyberGIS-Compute. We would also like to thank the many alpha and beta testers in the HydroShare and I-GUIDE communities who helped us to refine and improve the software.

References

- Wang S. A CyberGIS framework for the synthesis of cyberinfrastructure, GIS, and spatial analysis. *Ann Assoc Am Geogr* 2010;100(3):535–57. <https://doi.org/10.1080/00045601003791243>. Jun.
- Merkel D. Docker: lightweight linux containers for consistent development and deployment. *Linux J* 2014;2014(239):2.
- Kurtzer GM, Sochat V, Bauer MW. Singularity: scientific containers for mobility of compute. *PLoS ONE* 2017;12(5):e0177459. <https://doi.org/10.1371/journal.pone.0177459>. May.
- Apache Foundation (2024). Apache Airflow, v2.8.2 [Computer software]. Github. Retrieved March 4, 2024, from <https://github.com/apache/airflow>.
- Baker M. 1,500 scientists lift the lid on reproducibility. *Nature* 2016;533(7604). <https://doi.org/10.1038/533452a>. Art. no. 7604, May.
- Horsburgh JS, et al. HydroShare: sharing diverse environmental data types and models as social objects with application to the hydrology domain. *JAWRA J Am Water Resour Assoc* 2016;52(4):873–89. <https://doi.org/10.1111/1752-1688.12363>.
- Shook E, Vento DD, Zonca A, Wang J. GISandbox: a science gateway for geospatial computing. In: *Proceedings of the Practice and Experience on Advanced Research Computing*, in PEARC '18. New York, NY, USA: Association for Computing Machinery; 2018. p. 1–7. <https://doi.org/10.1145/3219104.3219150>. Jul.
- Yin D, et al. CyberGIS-Jupyter for reproducible and scalable geospatial analytics. *Concurr Comput Pract Exp* 2019;31(11):e5040. <https://doi.org/10.1002/cpe.5040>.
- Calyam P, et al. Measuring success for a future vision: defining impact in science gateways/virtual research environments. *Concurr Comput Pract Exp* 2021;33(19):e6099. <https://doi.org/10.1002/cpe.6099>.
- Shook E, Bowlick F, Padmanabhan A, Kemp K. The hour of cyberinfrastructure (hour of CI): early findings from pilot study to build cyber literacy for GIScience. In: 2021 IEEE/ACM Ninth Workshop on Education for High Performance Computing (EduHPC); 2021. p. 9–15. <https://doi.org/10.1109/EduHPC54835.2021.00007>. Nov.
- Yin D, Liu Y, Padmanabhan A, Terstriep J, Rush J, Wang S. A CyberGIS-Jupyter framework for geospatial analytics at scale. In: *Proceedings of the Practice and Experience in Advanced Research Computing 2017 on Sustainability, Success and Impact*. New York, NY, USA: Association for Computing Machinery; 2017. p. 1–8. <https://doi.org/10.1145/3093338.3093378>. in PEARC17Jul.
- Michels A, Padmanabhan A, Li Z, Wang S. Towards reproducible research on CyberGISX with Lmod and easybuild. *Zenodo* 2021. <https://doi.org/10.5281/zenodo.5569659>. Oct. 14.
- Michels A, Padmanabhan A, Li Z, Wang S. EasyScienceGateway: a new framework for providing reproducible user environments on science gateways. *Concurr Comput Pract Exper* 2024;36(4):e7929. <https://doi.org/10.1002/cpe.7929>.
- Boerner T, Deems S, Furlani T, Knuth S, Towns J. ACCESS: advancing innovation: NSF's advanced cyberinfrastructure coordination ecosystem: services & support. In: *Proceedings of the Practice and Experience on Advanced Research Computing*. Association for Computing Machinery; July 2023. p. 173–6. <https://doi.org/10.1145/3569951.3597559>. PEARC '23.
- Padmanabhan A, Xiao Z, Vandewalle R, Michels A, Wang S. Enabling computationally intensive geospatial research on CyberGIS-Jupyter with CyberGIS-compute. *Zenodo* 2021;14. <https://doi.org/10.5281/zenodo.5570056>.
- Padmanabhan A, et al. CyberGIS-compute for enabling computationally intensive geospatial research. In: *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on APIs and Libraries for Geospatial Data Science*. Association for Computing Machinery; Nov. 2021. p. 1–2. <https://doi.org/10.1145/3486189.3490017>. in SpatialAPI '21.
- Kluyver T, et al. Jupyter notebooks – a publishing format for reproducible computational workflows. In: Loizides F, Schmidt B, editors. *Positioning and power in academic publishing: players, agents and agendas*; 2016. p. 87–90.
- Honggao L, Wang S, Perez L, Chakravorty D, Cockerill T. ACES - accelerating computing for emerging sciences. *Comput Syst* 2023. <https://hprc.tamu.edu/aces/>.
- Yoo AB, Jette MA, Grondona M. SLURM: simple linux utility for resource management. In: Feitelson D, Rudolph L, Schwiiegelshohn U, editors. *Job scheduling strategies for parallel processing*. Berlin, Heidelberg: Springer; 2003. p. 44–60. https://doi.org/10.1007/10968987_3. Lecture Notes in Computer Science.
- OpenJS Foundation (2024). Express, v4.18.2 [Computer software]. Github. Retrieved March 4, 2024, from <https://github.com/expressjs/express>.
- Foster I, Kesselman C. Globus: a metacomputing infrastructure toolkit. *Int J Supercomput Appl High Perform Comput* 1997;11(2):115–28. <https://doi.org/10.1177/109434209701100205>. Jun.
- Kang J-Y, Farkhad BF, Chan MS, Michels A, Albarracín D, Wang S. Spatial accessibility to HIV testing, treatment, and prevention services in Illinois and Chicago, USA. *PLoS ONE* 2022;17(7):e0270404. <https://doi.org/10.1371/journal.pone.0270404>. Jul.
- Park J, Michels A, Lyu F, Han SY, Wang S. Daily changes in spatial accessibility to ICU beds and their relationship with the case-fatality ratio of COVID-19 in the state of Texas, USA. *Appl Geogr* 2023;102929. <https://doi.org/10.1016/j.apgeog.2023.102929>. Mar.
- Kang J-Y, Lee S. Exploring food deserts in Seoul, South Korea during the COVID-19 pandemic (from 2019 to 2021). *Sustainability* 2022;14(9). <https://doi.org/10.3390/su14095210>. Art. no. 9, Jan.
- Park J, Kang J-Y, Goldberg DW, Hammond TA. Leveraging temporal changes of spatial accessibility measurements for better policy implications: a case study of electric vehicle (EV) charging stations in Seoul, South Korea. *Int J Geogr Inf Sci* 2021;1–20. <https://doi.org/10.1080/13658816.2021.1978450>. Sep.
- Wang S. CyberGIS and spatial data science. *GeoJournal* 2016;81(6):965–8.
- Strande S, et al. Expanse: computing without boundaries: architecture, deployment, and early operations experiences of a supercomputer designed for the rapid evolution in science and engineering. Practice and experience in advanced research computing, in pearc '21. New York, NY, USA: Association for Computing Machinery; Jul. 2021. p. 1–4. <https://doi.org/10.1145/3437359.3465588>.
- Song C, et al. Anvil - system architecture and experiences from deployment and early user operations. Practice and experience in advanced research computing, in pearc '22. New York, NY, USA: Association for Computing Machinery; Jul. 2022. p. 1–9. <https://doi.org/10.1145/3491418.3530766>.
- Kang J-Y, et al. Rapidly measuring spatial accessibility of COVID-19 healthcare resources: a case study of Illinois, USA. *Int J Health Geogr* 2020;19(1):1–17.
- Haklay M, Weber P. OpenStreetMap: user-generated street maps. *IEEE Pervas Comput Oct*. 2008;7(4):12–8. <https://doi.org/10.1109/MPRV.2008.80>.
- Boeing G. OSMnx: new methods for acquiring, constructing, analyzing, and visualizing complex street networks. *Comput Environ Urban Syst* 2017;65:126–39. <https://doi.org/10.1016/j.compenurbsys.2017.05.004>.
- Luo W, Qi Y. An enhanced two-step floating catchment area (E2SFCA) method for measuring spatial accessibility to primary care physicians. *Health Place* 2009;15(4):1100–7. <https://doi.org/10.1016/j.healthplace.2009.06.002>.
- Lyu F, Yang Z, Xiao Z, Diao C, Park J, Wang S. CyberGIS for scalable remote sensing data fusion. In: *Practice and Experience in Advanced Research Computing*. New York, NY, USA: Association for Computing Machinery; Jul. 2022. p. 1–4. <https://doi.org/10.1145/3491418.3535145>. in PEARC '22.
- Li Z, Michels A, Padmanabhan A, Nassar A, Tarboton DG, Wang S. CyberGIS-Jupyter for water - an open geospatial computing platform for collaborative water research. In: *AGU Fall Meeting 2022*, December; 2022. <https://ui.adsabs.harvard.edu/abs/2022AGUFMIN32A.05L>.
- Institute for geospatial understanding through an integrative discovery environment (I-GUIDE). I-GUIDE Platform 2024. <https://iguide.illinois.edu/platform/>. 2023 [accessed 6 March].
- Institute for geospatial understanding through an integrative discovery environment (I-GUIDE). I-GUIDE VCO: CyberGIS-Compute: Geospatial Middle Simplify Access High-Perform Comput 2022. <https://iguide.illinois.edu/i-guide-vco/virtual-consulting-office-cybergis-compute/> [accessed 6 March 2024].
- Institute for geospatial understanding through an integrative discovery environment (I-GUIDE). I-GUIDE VCO: CyberGIS-Compute: Enabl Simplif Access High Perform Comput Geospat Comput 2022. <https://iguide.illinois.edu/i-guide-vco/i-guide-vco-cybergis-compute-part-2/> [accessed 6 March 2024].
- Institute for Geospatial Understanding through an Integrative Discovery Environment (I-GUIDE). In: *AAG 2023 Symposium on Harnessing the Geospatial Data Revolution for Sustainability Solutions*; 2023. <https://iguide.illinois.edu/aag-2023-symposium-on-harnessing-the-geospatial-data-revolution-for-sustainability-solutions/> [accessed 6 March 2024].
- Texas A&M high performance research computing. In: *ACES Workshop*; 2023. https://hprc.tamu.edu/aces/aces_workshop.html [accessed 6 March 2024].
- Baig F, et al. CyberGIS-cloud: a unified middleware framework for cloud-based geospatial research and education. Practice and experience in advanced research computing. New York, NY, USA: Association for Computing Machinery; 2022. p. 1–4. <https://doi.org/10.1145/3491418.3535148>. in PEARC '22Jul.

- [41] Borthakur D. The hadoop distributed file system: architecture and design. *Hadoop Proj Website* 2007;11(2007):21.
- [42] M. Zaharia, M. Chowdhury, M.J. Franklin, S. Shenker, and I. Stoica, "Spark: cluster computing with working sets," in *2nd USENIX workshop on hot topics in cloud computing (HotCloud 10)*, 2010.
- [43] Rocklin M. Dask: parallel computation with blocked algorithms and task scheduling. In: *Proceedings of the 14th Python in Science Conference (SciPy 2015)*; 2015.