Providing Accessible Software Environments Across Science Gateways and HPC

Alexander Michels
michels9@illinois.edu
University of Illinois
Urbana-Champaign
Urbana, Illinois, USA

Mit Kotak
mkotak@mit.edu
Massachusetts Institute of Technology
Cambridge, Massachusetts, USA

Anand Padmanabhan
apadmana@illinois.edu
University of Illinois
Urbana-Champaign
Urbana, Illinois, USA

John Speaks
jspeaks2@illinois.edu
University of Illinois
Urbana-Champaign
Urbana, Illinois, USA

Shaowen Wang
shaowen@illinois.edu
University of Illinois
Urbana-Champaign
Urbana, Illinois, USA

ABSTRACT

While High-Performance Computing (HPC) resources are powerful for tackling complex, computationally intensive analysis and modeling problems, access to these resources varies across disciplines. Domain scientists in a variety of fields such as social and environmental sciences often lack in-depth technical skills (e.g., familiarity with terminal, knowledge of job schedulers) to effectively utilize HPC resources, hindering desired research. In this context, CyberGIS-Compute is a middleware toolkit designed to democratize HPC access with the main goal of enabling domain scientists in diverse fields to solve computationally intensive problems. A key challenge facing model developers on CyberGIS-Compute is to create a containerized software environment for their models. Domain experts unfamiliar with HPC are generally unfamiliar with containerization technologies (e.g., Docker, Singularity) and thus unable to create/test containers to execute their models. But if they have access to science gateways, they would want to use these familiar software environments on HPC resources. This paper describes a novel approach to integrating the Cern Virtual Machine File System (CVMFS) into CyberGIS-Compute to provide consistent software environments across science gateways and HPC resources.

CCS CONCEPTS

• **Applied computing** → **Earth and atmospheric sciences**; • **Computing methodologies** → **Distributed computing methodologies**.

KEYWORDS

cyberGIS, HPC, middleware, Jupyter, Python, CVMFS

ACM Reference Format:
Alexander Michels, Mit Kotak, Anand Padmanabhan, John Speaks, and Shaowen Wang. 2024. Providing Accessible Software Environments Across Science Gateways and HPC. In *Practice and Experience in Advanced Research Computing (PEARC '24), July 21–25, 2024, Providence, RI, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3626203.3670614

PEARC '24, July 21–25, 2024, Providence, RI, USA

ACM ISBN 979-8-4007-0419-2/24/07
https://doi.org/10.1145/3626203.3670614

1 INTRODUCTION

Providing consistent software environments for scientific reproducibility is a continuing challenge in geospatial science and many related domains [16]. Science gateways [11], like CyberGISX [23] CyberGIS-Jupyter for Water (CJW) [9], and HydroShare [3], have made great strides towards providing executable environments and repositories for sharing computational workflows. While HPC resources are critical for executing computationally intensive workflows, many in geospatial science are not familiar enough with HPC to utilize them. Skills like using the terminal and SSH, and programming outside of an Integrated Development Environment (IDE) are not part of the typical curriculum in geospatial fields, resulting in a steep learning curve for utilizing HPC.

CyberGIS-Compute [17, 18] is designed to alleviate many of these issues: it is a middleware tool for executing models on HPC resources from a Jupyter-based graphical user interface (GUI). The model contribution process was also streamlined to encourage contributions from domain experts who only need to provide a Github repo, a short JSON manifest describing the model and its computational requirements, and a container to ensure the model has an appropriate software environment to execute in. Model execution leverages containerization technology to provide consistent execution environments across HPC resources [8]. While this has drastically lowered the barriers to utilizing HPC for domain experts, the reliance on containers is still a significant technical hurdle inhibiting model developers' productivity. Models used in CyberGIS-Compute often have specific software version dependencies to ensure reproducibility. However, many model developers are not familiar with containerization technology, and thus are unable to develop a container and test that their model executes correctly within a container.

To address the aforementioned challenge, this research aims to ensure that code which executes on one of the science gateways deployed with CyberGIS-Compute can then be run on an HPC resource with the exact same software stack. This would allow model developers to run and test small portions of

their workflows on our Jupyter-based [7] science gateways and remove the complexity of containers as a barrier to model contribution. With a container-based software environment on science gateways, this is trivial due to using the same container on HPC, but our science gateways rely on the EasyScienceGateway framework [16] to increase the reproducibility and portability of the software environment. This approach utilizes a CVMFS repository to provide users with the necessary software meaning there is no single container image containing the software environment. In this paper, we discuss our approach integrating the Cern Virtual Machine File System (CVMFS) [2] into CyberGIS-Compute with the goal of providing users with a consistent software environment across science gateways and HPC resources.

2 BACKGROUND

2.1 CyberGIS-Compute

CyberGIS [22], defined as cyberinfrastructure-based geographic information science and systems, has fueled research in fields like economics [13], emergency management [20], and public health [5, 14]. Advances of CyberGIS and HPC are not spread evenly throughout geospatial science though: the majority still primarily use personal computers for their computation [21]. CyberGIS-Compute is designed to democratize access to HPC resources by reducing the barriers facing domain experts. CyberGIS-Compute has been utilized to tackle challenges in hydrology [11], public health [6], and remote sensing [10].

CyberGIS-Compute has two main components: a Software Development Kit (SDK) that provides a Graphical User Interface (GUI) for end-users on Jupyter-based science gateways and a Core server that manages model execution on HPC resources [17]. The SDK is written in Python and the GUI built on Jupyter widgets because we found Python and Jupyter notebooks [7] were familiar to researchers in geospatial science. An earlier iteration of the SDK provided only Python functions for model execution, but the GUI was developed to further reduce the learning curve for using CyberGIS-Compute. When a user submits jobs using the SDK, the request is sent to the Core server's Application Programming Interface (API), which authenticates the user using a Jupyter token, transfers the model code and any input data, submits and monitors the job, and returns any results to the user using Globus when the job completes [4].

A key design goal of CyberGIS-Compute is portability: models should be able to run on a variety of HPC resources. To achieve this goal, CyberGIS-Compute executes models within Singularity containers [8]. While containers allow us to easily run models on a variety of HPC resources, they are also one of the largest technical barriers for model developers. Those unfamiliar with HPC are usually not familiar with containerization concepts and unable to create or test a container for the model. We have created Singularity images that closely mimic the software environment provided by kernels on the CyberGISX [15, 23] and CyberGIS-Jupyter for Water (CJW) [9], but manually recreating software environments is not scalable and error prone. We have also worked directly with model developers to create containers for their models, but achieving a working and tested container can be a time-intensive process and communicating the exact specifications for a software environment is challenging. Model developers are generally familiar with the software environments on the science gateways supported by CyberGIS-Compute and want to use the exact same software environment on HPC.

2.2 Cern Virtual Machine File System (CVMFS)

CVMFS is a distributed file system for distributing software efficiently on HPC systems [2]. CVMFS utilizes Content-Addressable Storage which provides content de-duplication and enables fast data integrity verification. Additionally, the distributed approach is designed for scalability and fault-tolerance. Software is written to a Stratum 0 server, Stratum 1 servers are geographically distributed read-only copies of Stratum 0, and a network of proxies provide end-users with access.

The CyberGISX and CJW science gateways have recently adopted the EasyScienceGateway framework [16] using the CVMFS repository illustrated in Figure 1. While this approach solves some of the problems faced by those science gateways, it also complicates the process of recreating their software environments. There is no longer a single container that can be easily added to CyberGIS-Compute to recreate the software environment. Further, while CVMFS is designed for software on HPC, not all HPC systems use CVMFS or allow users to add their own CVMFS repositories.

3 METHODS

A new Connector[1] was developed for CyberGIS-Compute to utilize CVMFS for model execution on HPC. In CyberGIS-Compute's architecture, a Connector is an interface between the Core server and HPC systems that implement functionalities like file transfers and interacting with the job submission system on HPC. While this work would provide model developers with the software available on our science gateways, the eventual goal is to eliminate the need for model developers to create Singularity containers entirely, instead installing all software as modules into a CVMFS repository which could be used anywhere. This has the potential to drastically reduce the technical barriers to model contribution: instead of trying to explain the concepts behind containerization, we can simply tell model experts that if their model runs on one of our science gateways, we can run it on HPC. Models still technically execute in a container to standardize paths and ensure we have the necessary software for interacting with the software in CVMFS (e.g. Lmod [12]), but the same container is used for all models. Instead of specifying a container, model developers are asked to specify a kernel from one of our science gateways for the model to run on.

The CVMFS Connector relies on `singcvmfs`[2] to provide the CVMFS software on HPC centers without requiring that administrators add our CVMFS repository. The `singcvmfs` tool is a drop-in replacement for the Singularity command that allows us to launch a Singularity container with CVMFS repositories bind-mounted in them. This provides the CVMFS repositories in our containers on HPC, but users cannot be expected to know which modules and executables are associated with each kernel, so our next step was to streamline the process of recreating the software environments

[1] https://github.com/cybergis/cybergis-compute-core/pull/64
[2] https://github.com/cvmfs/cvmfsexec

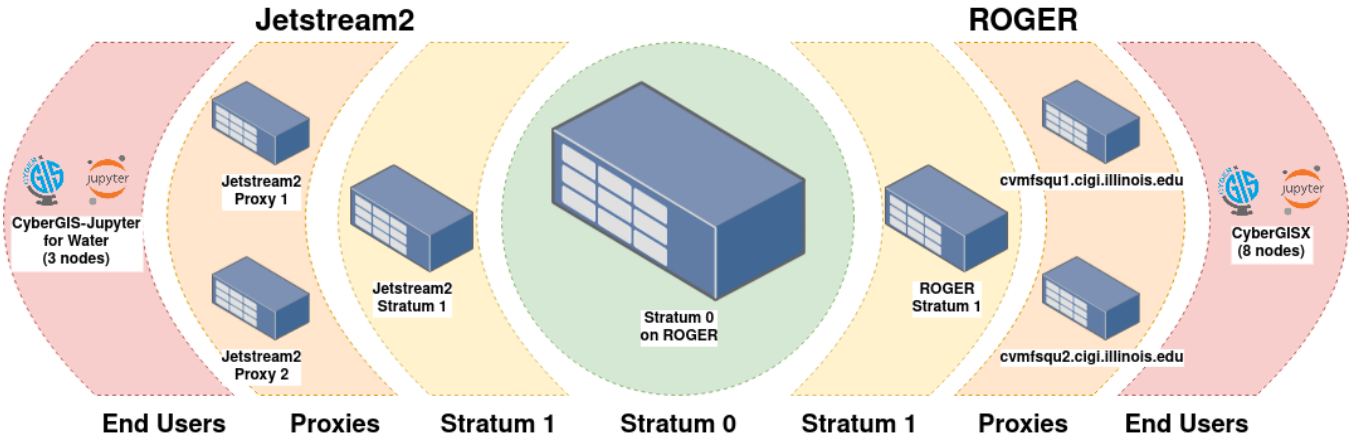

Figure 1: The CVMFS content delivery network used by the `cybergis.illinois.edu` CVMFS repository which serves the CyberGISX and CJW science gateways.

```
{
    "name": "Pysal Access Example",
    "description": "Calculates spatial accessibil
ity using a variety of metrics using the Pysal ac
cess package: https://github.com/pysal/access",
    "estimated_runtime": "3-6 minutes",
    "container": "cybergisx/python3-0.9.0",
    "connector": "SingCVMFSConnector",
    "execution_stage": "python ChicagoAccess.py",
    "slurm_input_rules": {
          "time": {
               "max": 30,
               "min": 15,
               "default_value": 20,
               "step": 1,
               "unit": "Minutes"
          },
          "memory": {
               "max": 4,
               "min": 2,
               "default_value": 4,
               "step": 1,
               "unit": "GB"
          }
    },
    "require_upload_data": false,
    "supported_hpc": ["anvil_community", "expanse
_community", "keeling_community"],
    "default_hpc": "keeling_community"
}
```

```
{
    "name": "Pysal Access Example",
    "description": "Calculates spatial accessibil
ity using a variety of metrics using the Pysal ac
cess package: https://github.com/pysal/access",
    "estimated_runtime": "3-6 minutes",
    "container": "pysal-access",

    "execution_stage": "python ChicagoAccess.py",
    "slurm_input_rules": {
          "time": {
               "max": 30,
               "min": 15,
               "default_value": 20,
               "step": 1,
               "unit": "Minutes"
          },
          "memory": {
               "max": 4,
               "min": 2,
               "default_value": 4,
               "step": 1,
               "unit": "GB"
          }
    },
    "require_upload_data": false,
    "supported_hpc": ["anvil_community", "expanse
_community", "keeling_community"],
    "default_hpc": "keeling_community"
}
```

Figure 2: Comparison of the manifests for the CVMFS-enabled model (left) and the original model (right). Only two lines have changed: a new line specifies the SingCVMFS-Connector and the container is changed from "pysal-access" to "cybergisx/python3-0.9.0."

from our science gateways on HPC. To make the process more user friendly, we configured our CVMFS Connector to generate a `kernel-init.sh` script corresponding to each kernel on our science gateway which recreates the software environment used by the kernel specified[3].

4 RESULTS

We successfully converted and executed existing models with our new CVMFS Connector. An example is the `pysal-acess` model[4] that uses the Pysal `access` Python package [19] to compare various methods of calculating spatial accessibility to doctors in Chicago, IL, USA. Crucially, converting the model to use our CVMFS Connector [5] required minimal changes: specifying that we want to use our new connector and the CyberGISX Python3 0.9.0 kernel in the container field. This is a small change to the model's manifest only and means we no longer need to design and maintain an additional container. Figure 2 shows a side-by-side comparison of the manifests for the CVMFS-enabled model (left) and the original model (right).

5 CONCLUDING DISCUSSION

This paper discusses our work to integrate CVMFS into CyberGIS-Compute, streamlining the process for model development and drastically reducing technical barriers facing domain experts trying to utilize advanced cyberinfrastructure. Our CVMFS Connector is demonstrated by executing an existing CyberGIS-Compute model and the conversion is accomplished with minimal changes to the metadata of the model. This removes the largest remaining technical barrier for model developers using CyberGIS-Compute and makes the software environments on our science gateways available on HPC resources. This new connector will also support a more diverse set of workflows which will no longer require additional containers: software environments in existing CVMFS repositories could be easily added to CyberGIS-Compute using this process.

This work demonstrates that our approach can work, but more work is needed to ensure that our solution is flexible, stable and reliable. CVMFS's cache works best when it is not on a shared filesystem, but many HPC systems do not provide significant storage on a shared filesystem resulting in sub-optimal performance. We have reported the issue to the `singcvmfs` developers[6] and will work with HPC administrators to determine the best path forward. Our CVMFS Connector occasionally causes failures if a model tries to access software not yet received from the content delivery network and we are working to minimize the frequency of such errors. Additionally, we have focused our work on Python-based models because they are the majority of CyberGIS-Compute models, but we need to extensively test if models in other languages or that

[3] Our kernel configuration can be found here: https://github.com/cybergis/cybergis-compute-core/blob/bd5fe96dbbdb7ec3f07c2ec5391645ca4a79104d/configs/kernel.example.json
[4] Original Pysal Access model: https://github.com/cybergis/pysal-access-compute-example
[5] CVMFS-enabled Pysal Access model: https://github.com/cybergis/pysal-access-compute-example-cvmfs
[6] https://github.com/cvmfs/cvmfsexec/issues/69

utilize software like MPI will work with our Connector. Our intial focus has been on HPC, but further work will explore running workflows on commercial clouds [1]. Lastly, outreach and further technical work will continue to lower technical barriers to accessing CyberGIS-Compute and support a wider variety of use-cases.

ACKNOWLEDGMENTS

This paper and associated materials are based in part upon work supported by the National Science Foundation under grant numbers: 2118329, 2112356, and 2321070. Our computational experiments used ROGER that is a geospatial supercomputer supported by the CyberGIS Center for Advanced Digital and Spatial Studies and the School of Earth, Society and Environment at the University of Illinois Urbana-Champaign.

REFERENCES

[1] Furqan Baig, Alexander Michels, Zimo Xiao, Su Yeon Han, Anand Padmanabhan, Zhiyu Li, and Shaowen Wang. 2022. CyberGIS-Cloud: A Unified Middleware Framework for Cloud-Based Geospatial Research and Education. In *Practice and Experience in Advanced Research Computing (PEARC '22)*. Association for Computing Machinery, New York, NY, USA, 1–4. https://doi.org/10.1145/3491418.3535148

[2] Jakob Blomer, Predrag Buncic, and Thomas Fuhrmann. 2011. CernVM-FS: Delivering Scientific Software to Globally Distributed Computing Resources. In *Proceedings of the First International Workshop on Network-aware Data Management - NDM '11*. ACM Press, Seattle, Washington, USA, 49. https://doi.org/10.1145/2110217.2110225

[3] Young-Don Choi, Jonathan L. Goodall, Jeffrey M. Sadler, Anthony M. Castronova, Andrew Bennett, Zhiyu Li, Bart Nijssen, Shaowen Wang, Martyn P. Clark, Daniel P. Ames, Jeffery S. Horsburgh, Hong Yi, Christina Bandaragoda, Martin Seul, Richard Hooper, and David G. Tarboton. 2021. Toward Open and Reproducible Environmental Modeling by Integrating Online Data Repositories, Computational Environments, and Model Application Programming Interfaces. *Environmental Modelling & Software* 135 (Jan. 2021), 104888. https://doi.org/10.1016/j.envsoft.2020.104888

[4] Ian Foster and Carl Kesselman. 1997. Globus: A Metacomputing Infrastructure Toolkit. *The International Journal of Supercomputer Applications and High Performance Computing* 11, 2 (June 1997), 115–128. https://doi.org/10.1177/109434209701100205

[5] Jeon-Young Kang, Bita Fayaz Farkhad, Man-pui Sally Chan, Alexander Michels, Dolores Albarracin, and Shaowen Wang. 2022. Spatial Accessibility to HIV Testing, Treatment, and Prevention Services in Illinois and Chicago, USA. *PLOS ONE* 17, 7 (July 2022), e0270404. https://doi.org/10.1371/journal.pone.0270404

[6] Jeon-Young Kang, Alexander Michels, Fangzheng Lyu, Shaohua Wang, Nelson Agbodo, Vincent L Freeman, and Shaowen Wang. 2020. Rapidly Measuring Spatial Accessibility of COVID-19 Healthcare Resources: A Case Study of Illinois, USA. *International journal of health geographics* 19, 1 (2020), 1–17.

[7] Thomas Kluyver, Benjamin Ragan-Kelley, Fernando Pérez, Brian Granger, Matthias Bussonnier, Jonathan Frederic, Kyle Kelley, Jessica Hamrick, Jason Grout, Sylvain Corlay, Paul Ivanov, Damián Avila, Safia Abdalla, and Carol Willing. 2016. Jupyter Notebooks – a Publishing Format for Reproducible Computational Workflows. In *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, F. Loizides and B. Schmidt (Eds.). IOS Press, IOS Press, Göttingen, Germany, 87–90. https://doi.org/10.3233/978-1-61499-649-1-87

[8] Gregory M. Kurtzer, Vanessa Sochat, and Michael W. Bauer. 2017. Singularity: Scientific Containers for Mobility of Compute. *PLOS ONE* 12, 5 (May 2017), e0177459. https://doi.org/10.1371/journal.pone.0177459

[9] Zhiyu Li, Alexander Michels, Anand Padmanabhan, Ayman Nassar, David G. Tarboton, and Shaowen Wang. 2022. CyberGIS-Jupyter for Water - an Open Geospatial Computing Platform for Collaborative Water Research. In *AGU Fall Meeting Abstracts*, Vol. 2022. The SAO/NASA Astrophysics Data System, Chicago, IL, IN32A–05.

[10] Fangzheng Lyu, Zijun Yang, Zimo Xiao, Chunyuan Diao, Jinwoo Park, and Shaowen Wang. 2022. CyberGIS for Scalable Remote Sensing Data Fusion. In *Practice and Experience in Advanced Research Computing (PEARC '22)*. Association for Computing Machinery, New York, NY, USA, 1–4. https://doi.org/10.1145/3491418.3535145

[11] Iman Maghami, Ashley Van Beusekom, Lauren Hay, Zhiyu Li, Andrew Bennett, YoungDon Choi, Bart Nijssen, Shaowen Wang, David Tarboton, and Jonathan L. Goodall. 2023. Building Cyberinfrastructure for the Reuse and Reproducibility of Complex Hydrologic Modeling Studies. *Environmental Modelling & Software* 164 (June 2023), 105689. https://doi.org/10.1016/j.envsoft.2023.105689

[12] Robert McLay, Karl W. Schulz, William L. Barth, and Tommy Minyard. 2011. Best Practices for the Deployment and Management of Production HPC Clusters. In *State of the Practice Reports (SC '11)*. Association for Computing Machinery, New York, NY, USA, 1–11. https://doi.org/10.1145/2063348.2063360

[13] Alexander Michels, Jeon-Young Kang, and Shaowen Wang. 2020. An Exploration of the Effect of Buyer Preference and Market Composition on the Rent Gradient Using the ALMA Framework. In *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on GeoSpatial Simulation (GeoSim '20)*. Association for Computing Machinery, New York, NY, USA, 48–51. https://doi.org/10.1145/3423335.3428167

[14] Alexander Michels, Jeon-Young Kang, and Shaowen Wang. 2022. Particle Swarm Optimization for Calibration in Spatially Explicit Agent-Based Modeling. *Journal of Artificial Societies and Social Simulation* 25, 2 (2022), 8.

[15] Alexander Michels, Anand Padmanabhan, Zhiyu Li, and Shaowen Wang. 2021. Towards Reproducible Research on CyberGISX with Lmod and Easybuild. https://doi.org/10.5281/zenodo.5569659

[16] Alexander Michels, Anand Padmanabhan, Zhiyu Li, and Shaowen Wang. 2023. EasyScienceGateway: A New Framework for Providing Reproducible User Environments on Science Gateways. *Concurrency and Computation: Practice and Experience* 36, 4 (2023), e7929. https://doi.org/10.1002/cpe.7929

[17] Alexander C. Michels, Anand Padmanabhan, Zimo Xiao, Mit Kotak, Furqan Baig, and Shaowen Wang. 2024. CyberGIS-Compute: Middleware for Democratizing Scalable Geocomputation. *SoftwareX* 26 (May 2024), 101691. https://doi.org/10.1016/j.softx.2024.101691

[18] Anand Padmanabhan, Ximo Ziao, Rebecca C. Vandewalle, Furqan Baig, Alexander Michels, Zhiyu Li, and Shaowen Wang. 2021. 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 (SpatialAPI '21)*. Association for Computing Machinery, New York, NY, USA, 1–2. https://doi.org/10.1145/3486189.3490017

[19] James Saxon, Julia Koschinsky, Karina Acosta, Vidal Anguiano, Luc Anselin, and Sergio Rey. 2022. An Open Software Environment to Make Spatial Access Metrics More Accessible. *Journal of Computational Social Science* 5, 1 (May 2022), 265–284. https://doi.org/10.1007/s42001-021-00126-8

[20] Rebecca Vandewalle, Jeon-Young Kang, Dandong Yin, and Shaowen Wang. 2019. Integrating CyberGIS-Jupyter and Spatial Agent-Based Modelling to Evaluate Emergency Evacuation Time. In *Proceedings of the 2nd ACM SIGSPATIAL International Workshop on GeoSpatial Simulation (GeoSim '19)*. Association for Computing Machinery, New York, NY, USA, 28–31. https://doi.org/10.1145/3356470.3365530

[21] Rebecca C. Vandewalle, William C. Barley, Anand Padmanabhan, Daniel S. Katz, and Shaowen Wang. 2021. Understanding the Multifaceted Geospatial Software Ecosystem: A Survey Approach. *International Journal of Geographical Information Science* 35, 11 (Nov. 2021), 2168–2186. https://doi.org/10.1080/13658816.2020.1831514

[22] Shaowen Wang. 2010. A CyberGIS Framework for the Synthesis of Cyberinfrastructure, GIS, and Spatial Analysis. *Annals of the Association of American Geographers* 100, 3 (June 2010), 535–557. https://doi.org/10.1080/00045601003791243

[23] Dandong Yin, Yan Liu, Hao Hu, Jeff Terstriep, Xingchen Hong, Anand Padmanabhan, and Shaowen Wang. 2019. CyberGIS-Jupyter for Reproducible and Scalable Geospatial Analytics. *Concurrency and Computation: Practice and Experience* 31, 11 (2019), e5040. https://doi.org/10.1002/cpe.5040

Received 27 April 2024; accepted 25 May 2024