Enabling High Performance Cloud Computing for the Community Multiscale Air Quality Model (CMAQ) version 5.3.3: Performance Evaluation and Benefits for the User Community

Christos I. Efstathiou[1][†], Elizabeth Adams[1], Carlie J. Coats[1], Robert Zelt[2], Mark Reed[2], John McGee[2], Kristen M. Foley[3], Fahim I. Sidi[3], David C. Wong[3], Steven Fine[4], Saravanan Arunachalam[1][*]

[1]Institute for the Environment, The University of North Carolina at Chapel Hill, Chapel Hill, 27599, USA
[2]Research Computing, Information Technology Services, The University of North Carolina at Chapel Hill, Chapel Hill, 27599, USA
[3]Center for Environmental Measurement and Modeling, Office of Research and Development, U.S. Environmental Protection Agency, Research Triangle Park, NC, USA
[4]Formerly with the Office of Air and Radiation, U.S. Environmental Protection Agency, Washington, DC, USA

[†]Currently with Physicians, Scientist, and Engineers for Healthy Energy, Oakland, CA, USA

Correspondence to: Saravanan Arunachalam (sarav@email.unc.edu)

Abstract. The Community Multiscale Air Quality (CMAQ) Model is a local-to-hemispheric scale numerical air quality modeling system developed by the U.S. Environmental Protection Agency (USEPA) and supported by the Center for Community Modeling and Analysis System (CMAS). CMAQ is used for regulatory purposes by the USEPA program offices and state and local air agencies and is also widely used by the broader global research community to simulate and understand complex air quality processes and for computational environmental fate and transport, and climate and health impact studies. Leveraging state-of-the-science cloud computing resources for high performance computing (HPC) applications, CMAQ is now available as a fully tested, publicly available technology stack (HPC cluster and software stack) for two major cloud service providers (CSPs). Specifically, CMAQ configurations and supporting materials have been developed for use on their HPC clusters, including extensive online documentation, tutorials, and guidelines to scale and optimize air quality simulations using their services. These resources allow modelers to rapidly bring together CMAQ, cloud-hosted datasets, and visualization and evaluation tools on ephemeral clusters that can be deployed quickly and reliably worldwide. Described here are considerations in CMAQv5.3.3 cloud use and the supported resources for each CSP, presented through a benchmark application suite that was developed as an example of typical simulation for testing and verifying components of the modeling system. The outcomes of this effort are to provide findings from performing CMAQ simulations on the cloud using popular vendor provided resources, to enable the user community to adapt this for their own needs and identify specific areas of potential optimization with respect to storage and compute architectures.

1 Introduction

Over the past decade, cloud computing has received a tremendous amount of attention for its potential to enable and simplify high performance computing (HPC) applications. Modeling user communities can greatly benefit by having real time access to cloud-ready reproducible workflows that include complex models and large datasets. Benefits can include reduced effort required to manage computational resources, the ability to rapidly obtain more resources when needed, flexible approaches for managing costs, and new opportunities for convenient data sharing. State-of-the-science numerical models simulating a variety of different processes and scales ranging from global circulation models to regional and high-resolution weather prediction workloads, have been demonstrated to perform efficiently on HPC infrastructure in the cloud. Development groups for earth system models such as weather, climate, ocean circulation, and air quality are currently designing and deploying modeling platforms or components that utilize different cloud environments (Campbell et al., 2023; Powers et al., 2021; Zhuang et al., 2020; Chui et al., 2019; Eastham et al., 2018; Chen et al., 2017).

The vast majority of such applications leverage "Infrastructure as a Code" (IaaC) or "Infrastructure as a Service" (IaaS) technologies and storage options provided by different cloud service providers which creates the need for a flexible approach in terms of data integration. In the context of air quality models, cloud computing encapsulates both the data storage and parallel computing requirements for large scale and high-resolution air quality simulations that frequently rely on output generated by other models that are dependent on chosen science configurations. Specifically, numerical models for simulating regional and global scale air quality events are developed with a core function to support a variety of science configuration options that are enabled at compile-time in addition to a suite of run-time options. Efforts needed to treat such complex system models in the software as a service (SaaS) paradigm (Zhang et al., 2019) have remained exploratory and not gained enough traction, as correctly applying such models to specific situations demands a level of user control that goes beyond what is considered "power user" and involves administrative skills and in-depth HPC knowledge. This makes model deployment extremely difficult to achieve through a web-based interface. While end users have the option to use images with precompiled standardized versions of air quality models through CSP marketplace offerings at an hourly cost, these commercial products are designed for specific implementations, and their associated base science options.

As an example, Zhuang et al., (2020) demonstrated the scalability of GEOS-Chem to thousands of cores using the AWS ParallelCluster to achieve similar computational and cost efficiencies of local HPC clusters. They provided an easy-to-follow research workflow in an HPC cluster environment on the cloud. We extended this work by running the CMAQ model on Amazon Web Services (AWS) ParallelCluster and Microsoft Azure CycleCloud and using the HPC Cluster high-level frameworks or IaaC provided by these two major cloud providers. We provide tutorials that give end-users the ability to reproducibly provision HPC clusters and software in a way that is optimized to run CMAQ on the cloud in a turn-key service.

65 Furthermore, the increase in availability of large datasets in the cloud through vehicles such as NOAA's Big Data Program/ NOAA Open Data Dissemination (Simonson et al., 2022; NOAA's Big Data Program, 2023), CMAS's Data Warehouse on AWS (CMAS's Data Warehouse on AWS, 2023), and GEOS-Chem registry of open data (GEOS-Chem registry of open data, 2023) is another incentive to develop cloud solutions for air quality models that provide more leverage to the end-user. Such initiatives are critical for the mission and growth of cloud modeling and CSPs have acknowledged and addressed the emerging

70 need of data democratization by waiving fees or providing free credits to facilitate access by scientists and average non-technical users of information systems. Tools such as AWS ParallelCluster and Azure CycleCloud are services that extend the power of IaaS by mimicking on-site HPC setups and provide an even more dynamically scalable environment that enables CMAQ modelers to step beyond the limits of single virtual machines (VMs), using the Simple Linux Utility Resource Management (Slurm) (Yoo et al, 2003) batch scheduler in a way that enables auto-scaling of the compute nodes, simplifying

75 the cluster deployment and management. It is important to emphasize, that ParallelCluster and Azure CycleCloud extend the capability from simply being able to run on a VM hosted in the cloud to a turn-key batch scheduling HPC tightly-coupled cluster environment that is dedicated to the end-user.

The Community Multiscale Air Quality (CMAQ) Model (Byun and Schere, 2006; Foley et al., 2010; Appel et al., 2017, 2021)

80 is an open-source modeling system that consists of a family of programs for conducting air quality simulations and is being actively developed. The Community Modeling and Analyses System (CMAS) Center facilitates community model development by hosting, developing and distributing software such as the CMAQ model, hosting the CMAS Center User Forum to facilitate exchange of information related to code and datasets and troubleshooting, and providing outreach and support through new user training, annual conferences and workshops on specific topics. In many cases, one or more factors

85 are increasing resource requirements for CMAQ simulations, including the addition of more complex algorithms to CMAQ, simulations of longer time periods or larger domains, and modeling grids with finer resolutions. For institutions that use traditional HPC centers, despite the evolution of job managers, resources frequently come with allocation time limits and long queue times. Even if groups can afford to acquire and maintain appropriate computing capacity, such an approach may not be cost-effective, especially if the capacity is not fully utilized. By leveraging cloud infrastructure, CMAQ users can pay monthly

90 on-demand fees to perform model simulations on clusters managed by commercial providers, without having to pay large up-front costs to purchase computer clusters or hire staff to maintain them. This can be extremely useful in enhancing computationally demanding research and air quality forecasting at an international scale, in many cases offering unprecedented expansion of such capabilities for developing nations. Another advantage of this approach is timely access to cutting edge processors, that otherwise would require disproportionate wait time, resources, and effort to obtain. Similarly, scalability can

95 be expanded in real-time and with minimal effort.

The purpose of this study is to demonstrate the efforts required to bring the CMAQ model version 5.3.3 (U.S. EPA, 2023) to the cloud and perform air quality simulations efficiently and affordably, leveraging existing and publicly available datasets. In the following sections, we describe several key aspects of this work:

100

- Develop benchmark test suites that can address and replicate the needs of a typical CMAQ user
- Streamline the CMAQ installation process in Amazon's AWS and Microsoft's Azure
- Demonstrate running CMAQ on the cloud and estimate associated costs, making suggestions on different options available to the modeling community
105
- Perform benchmark tests with different HPC clusters and their underlying VMs, networking, and storage options while keeping track of the performance and associated costs
- Make recommendations that would help reduce CMAQ simulation times specific to the cloud platform.
- Provide instructions for obtaining and using input datasets from the CMAS Data Warehouse under the AWS Open Data Program which waives data egress costs
110

The methodologies used in this study are available as hand-on tutorials, with details for a variety of HPC systems on different CSPs, guides, and recommendations for specific user needs (see the links under *HPC Cluster Deployment Options*).

2. CMAQ Workflow and Cloud Benchmark Suite

115 Air quality modeling systems such as CMAQ rely heavily on the parameterization and simulation output from numerical weather prediction (NWP) systems in an offline coupling manner facilitated by pre-processing tools. Initial and boundary conditions for regional-to-urban scale simulations can be defined by the user to be either static or the result of nested downscaling from a coarser domain model application (i.e., Hemispheric CMAQ). A common CMAQ workflow involves:

1. developing meteorological fields with the Weather Research and Forecasting (WRF) Model (Skamarock et al., 2021),

120 2. processing WRF output using the Meteorology-Chemistry Interface Processor (MCIP) (Otte et al, 2010),

3. developing emissions inputs using the Sparse Matrix Operator KErnel (SMOKE) modeling system (Houyoux et al, 2000),

4. developing other inputs such as initial and boundary conditions using preprocessors,

5. performing air quality simulations using the complete set of inputs,

6. assessing the successful completion of the simulation and verifying the model output, and

125 7. analysis of the results to address the purpose of the simulation (e.g., regulatory or research issues).

Cloud storage enables reproducible workflows by having both model and datasets publicly available and directly accessed by the run scripts.

Traditionally, every CMAQ release is distributed with a lightweight test case that includes all inputs necessary for the user to confirm a successful installation and completion of a multi-day simulation. Similarly, a newly standardized test case, referred to as the cloud benchmark suite (CBS), was developed to evaluate CMAQ's performance on cloud HPC environments. Benchmark suite simulations were designed considering different user needs and data availability to construct a well-established bundle of inputs and outputs that can be further scaled and customized to meet specific scalable requirements.

The hardware configuration necessary to run CMAQ depends on the domain size, grid resolution, complexity of physics and chemistry parameterization, number of variables and layers saved to the concentration and diagnostic files, and simulation duration. Since typical input and output data sets for CMAQ include three-dimensional descriptions of the dynamical and chemical state of the simulated atmosphere, these datasets could require several gigabytes of disk storage per simulation day.

Given these considerations, a two-day CBS for the contiguous United States (CONUS) was constructed with the aim to be representative of a commonly used domain over a time frame that can be used to fully test the CMAQ system. Typical requirements for a CONUS 12-km x12-km horizontal grid resolution are provided in Table 1 below, while Figure 1 shows a map with the domain's coverage.

All simulations used a modified version of the CMAQ model version 5.3.3 with the CB6 chemical mechanism and aerosol module 7 (cb6r3_ae7_aq). The Detailed Emissions Scaling, Isolation and Diagnostic (*DESID*) module was also used to reduce the emissions of a specific emission stream for a specific region within the domain to highlight this new emission scaling capability offered within CMAQ (Murphy, et al., 2021). Further details are provided later in this paper. Datasets are typically created in the NetCDF data format (Unidata, 2023) that allows for sharing on the cloud following programming methods that leverage the power of Models-3/EDSS Input/Output Applications Programming Interface (The BAMS/EDSS/Models-3 I/O API: User Manual, Related Programs, and Examples, 2023). Figure 2 shows a subset of the domain showing the reduction in concentration of NO_2 over the Northeastern US due to scaling the point source emissions from electric generating units (PT_EGU) in NY by 75%, for a base minus sensitivity case using the DESID module. This plot illustrates the resolution of the grid as compared to the state boundary lines. The storage space requirements are defined based on the need to perform multiple sets of identical runs while changing the number of cores used to run CMAQ for single node and parallel HPC clusters using OpenMPI (Gabriel, et al., 2004) to evaluate scalability and to accommodate the additional disk space required for base and sensitivity runs.

Benchmark runs were performed with two output options: the first using a fully enabled CONC output option (37 variables, 36 layers), the second with a reduced number of variables and layers saved to the output concentration (CONC file) (12

Deleted: eaern

Deleted: The storage space requirements are defined based on the need to perform multiple sets of identical runs while changing the number of CPUs used to run CMAQ in single -node workflows or parallel HPC implementations using OpenMPI (Gabriel, et al., 2004). …

Deleted: Additional benchmark simulations were performed using different MPI configurations to change the number of available processors in order to evaluate scalability. …

Formatted: Font: Italic

Formatted: Font: Not Italic

170 variables, 1 layer). The scaling benchmarks used the reduced file option because the I/O API in its current version is not parallelized, and using the full output file may have negatively impacted the compute portion of scaling.

Figure 3 shows the flowchart of how to run CMAQ with the Chemical Transport Model (CTM) scientific processes color coded to match the benchmark timing equivalent found in the main log file in Figures 8 - 12.

175

Table 1: CMAQ model configuration and storage needs for the CONUS case benchmark suite.

CMAQ version	5.3.3 with code modifications to fix cloud-specific bugs
OS	Linux, Processor 64-bit x86 (Ubuntu on AWS, Almalinux on Azure)
Memory	>1 GiB RAM per CPU core
Storage	Disk space requirement for the 2-day Benchmark Suite is 250 GB: 44 GB Input data and 170 GB Output data (output files included concentrations for all species, all layers) (CBS_full) or 18 GB Output data (for CONC file limited to 12 species, 1 layer) (CBS_limited)
Domain (ncols x nrows x nlays)	396 x 246 x 35
Horizontal domain resolution	12-km x 12-km
Temporal resolution of output	Hourly
Temporal duration	2 days
Chemical mechanism	cb6r3_ae7_aq

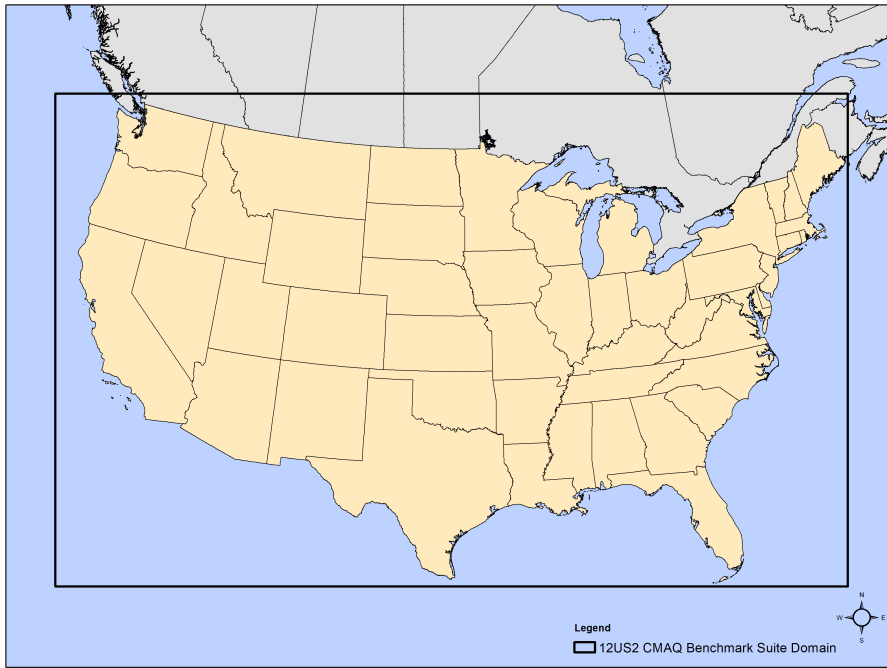
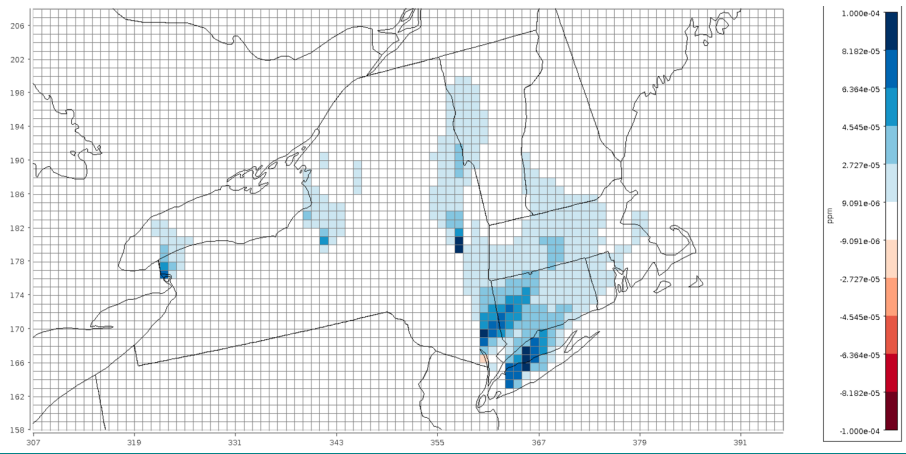
180

185

Figure 1: Cloud benchmark suite modeling domain ("12US2"; 396 columns × 246 rows × 35 vertical layers) for the CONUS at a 12-km x 12-km horizontal grid spacing is shown as the bold rectangle.

190   **Figure 2: Spatial Plot of differences in NO$_2$ concentrations when power plant emissions in NY were reduced by 25%. Lines represent 12-km x 12-km model grid while the x and y axes represent the row and column numbers of the model grid cells for the CONUS domain shown in Figure 1.**

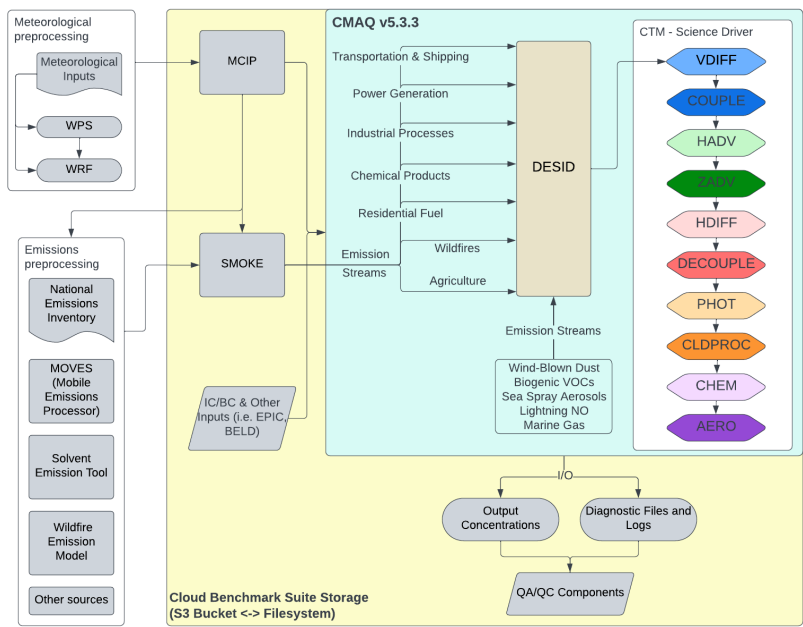



Figure 3: CMAQ Flowchart (CTM science driver processes are color coded to match timings captured in Figures 8-12).

3 CMAQ Experimental Design for CSPs

3.1 CMAQ Software Stack

Model and prerequisite libraries: Installing and setting up CMAQ on different CSPs with comparable Linux operating systems follows the general method depicted in the schematic of Figure 4. Step-by-step instructions to install the software stack using automated C-shell scripts are provided in the online tutorials. In addition, the tutorial covers the preparation of the benchmark data and provides run scripts for launching CMAQ through the job manager. To facilitate an even better approach, publicly available snapshots of the /shared volume that contains the software stack are provided for each CMAQ model and hardware

release. This allows new users to build clusters and quickly run CMAQ on HPC systems on the cloud. Additionally, it allows users to directly invoke existing precompiled libraries as modules, allowing for multiple applications and versions to be used and speed up model workflows and modifications (https://modules.readthedocs.io/en/latest/).

Depending on networking and storage options, users may need to add specific drivers and/or filesystem clients/layers to the list of installed modules. In parallel filesystem cases like Lustre, a client that is OS-specific needs to be present and linked to a storage account associated with the cluster to proceed for Azure Cycle Cloud. AWS offers a built-in Lustre implementation for most of their VMs including ParallelCluster. Azure VM images with embedded Lustre clients linked to a Lustre volume, currently in a public beta testing phase, were made available for our benchmark cases.

Data Transfer Options. AWS VMs have the AWS Command Line Interface (CLI) that is used to copy data from the S3 buckets available to the public through the AWS Open Data Program. For the case of Azure, users are provided instructions to install and use the AWS CLI and a *csh* script to copy the data from the CMAS Data Warehouse on AWS Open Data S3 bucket to the storage option being used. Data could also be copied from non-public S3 buckets to which the user has access privileges. An alternative is to link the S3 bucket to Lustre on AWS or create blob storage on Azure, and connect that blob storage to Lustre directly, to speed up access to input data. Azure users may want to use datasets from Microsoft's AI for Earth Data Sets (https://microsoft.github.io/AIforEarthDataSets/).

HPC Cluster Deployment Options. Step-by-step guidance for each CSP and the workflow used to run the benchmark has been documented and provided in the following tutorials (Azure: https://cyclecloud-cmaq.readthedocs.io/en/latest/ AWS: https://pcluster-cmaq.readthedocs.io/en/latest/). A verbose section was included in the run script structure to allow for recording architecture and OS specific parameters in the log files, including higher precision timing tracking of each model process. Recommendations for optimal MPI process placement using the Slurm Workload Manager with pinning on Azure HB series VMs were established for CycleCloud applications (https://techcommunity.microsoft.com/t5/azure-high-performance-computing/optimal-mpi-process-placement-for-azure-hb-series-vms/ba-p/2450663). Process placement was also used for ParallelCluster applications on AWS, optimized for the Hpc6a series. In the process outlined in Figure 4, we have also included code profiling tools (e.g., ARM MAP – https://www.linaroforge.com/) which allow for a better understanding of code performance and optimization opportunities for various applications/problem sizes. In Figures 5 and 6, we present overview schematics of the single VM and cluster configuration in each CSP. With respect to storage options, we chose the naming convention */local* to refer to running CMAQ and saving the output on a local SSD, */lustre* for performing the simulations on the Lustre parallel file system on AWS and Azure, */shared* for using Elastic Block Store (EBS) on AWS and built-in Network File System (NFS) volume with default configuration on Azure, and */data* for Azure's external NFS share option (for more information see: https://learn.microsoft.com/en-us/azure/storage/common/nfs-comparison and

https://docs.aws.amazon.com/parallelcluster/latest/ug/SharedStorage-v3.html). In general, storage implementations are CSP-specific and have different performance characteristics and fine-tuning options.

HPC Cluster Monitoring Options. The AWS CloudWatch (https://aws.amazon.com/cloudwatch) webpage interface was used to monitor and compare the throughput of the I/O on the EBS and Lustre file systems using full output 37 variables, and all layers in the CONC file. The Azure Monitor Metrics (https://learn.microsoft.com/en-us/azure/azure-monitor/essentials/data-platform-metrics) webpage interface was used to compare the latency and throughput of the I/O on the shared and Lustre file systems using the cloud benchmark suite (CBS_full).

```
┌─────────────────────────────────────┐  ↑
│       Post-processing tools         │  │
│      VERDI visualization tools      │  Step 3
│       R Language & packages         │
│         QA/QC scripts               │
├─────────────────────────────────────┤
│         CMAQ model                  │
│ Run scripts for different systems/  │  Step 2
│        configuration options        │
├─────────────────────────────────────┤
│    csh, git, and other basic tools  │
│        gcc/hpx compilers            │
│           netcdf                    │  Step 1
│            I/O API                  │
│           OpenMPI                   │
│            SLURM                    │  │
└─────────────────────────────────────┘
```

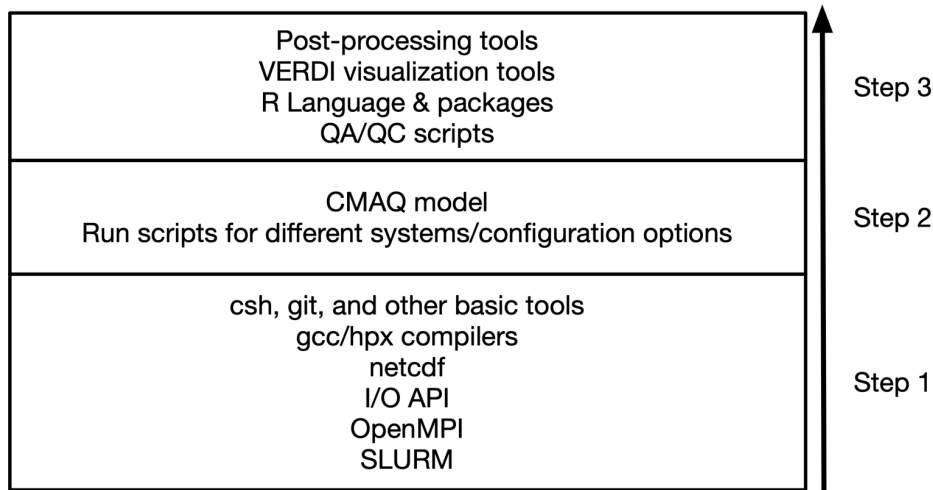



Figure 4: Multi-step approach in installing CMAQ prerequisites, model code, post-processing, and other visualization/evaluation tools of the software stack.

3.2 CSP Computation Options

The first step to begin using the cloud is to engage a cloud service provider (CSP) and create an account. This is the user's responsibility and CSPs have direct dedicated support to address specific user needs. Cloud-based CMAQ setups were developed and are currently available on two CSPs: Amazon (AWS) and Microsoft (Azure). Typical CMAQ modeling workflows on the cloud can be divided into two general approaches: provisioning a single virtual machine and provisioning a

dynamic multi-node cluster system. As multiprocessing architectures have evolved, many vendors are offering single VMs with more than 100 CPU cores, making them ideal for flexibly allocating and managing resources for computational simulations while limiting the effort required to compile and maintain the code/scripts. Clusters can be created in a multi-node framework following a similar approach, once access and availability of the total amount of resources is granted by the CSP.

270

After a thorough initial testing of the model code with a wide spectrum of hardware options offered by cloud vendors for HPC applications, we established the best performing architecture configurations described in Table 2 as the hardware stack test bed for final benchmarking under this study. Amazon's Hpc6a instances are powered by two 48-core 3rd generation AMD EPYC 7003 series processors built on 7nm process nodes for increased efficiency, 384 GiB of memory, 256 MB of L3 cache,

275 with a total of 96 cores. AWS Nitro System offloads the hypervisor to a dedicated card that also provides high speed networking and high-speed Elastic Block Store (EBS) services, allowing all 96 cores of the AMD chip to be used for simulations (AWS, 2023). Azure's HB120v3 server features two 64-core AMD EPYC 7V73X processors for a total of 128 physical cores, while each section contains 8 processor cores with uniform access to a 96 MB L3 cache/section. The Azure HB 120v3 was designed to reserve eight cores for the Hypervisor and provides the remaining 120 cores for the application. Modern processors such as

280 AMD's EPYC series employ Non-Uniform Memory Access, a multiprocessing (multi-die) architecture in which each processor is attached to its own local memory (called a NUMA domain) but can also access memory attached to another processor. To maximize the performance for each AMD chip it is important to balance the amount of L3 cache and memory bandwidth per core at the job level. This means that the binding of a process or thread to a specific core, known as CPU pinning or processor affinity, will now have to include additional steps for NUMA topology optimization (Podzimek et al., 2015;

285 Ghatrehsamani et al., 2020).

3.3 Networking Options

In the tutorials and code implementations, we employed CSP-specific advanced networking options that reflect the available

290 hardware options, enabling the 100 Gbps Elastic Fabric Adapter (EFA) on AWS, and the 200 Gbps High Dynamic Range (HDR) InfiniBand on Azure to support the level of high-performance computing required by CMAQ.

3.4 Storage Options

295 For storage, the real-time allocation of bandwidth and input/output operations per second (IOPS) differs between cloud vendors and should be examined independently at the application level by the user. In the examples investigated in this study, the user has access to four CSP-specific types of storage:

1. The fastest built-in local storage using Nonvolatile Memory Express Solid-State Drives (NVMe SSDs) that is included with default single VM provisioning

300 2. Network file systems tied to the user/enterprise account accessible using the Network File System (NFS) for Azure and Elastic File System (EFS) for AWS, attached to the head node in a cluster environment or directly to a VM

3. Unique services such as AWS's Elastic Block Store (EBS) which are designed for as per-instance block for certain compute cloud frameworks such as single AWS elastic cloud (EC2) and Azure's NetApp Files (ANF)

4. Fully managed high performance file system such as Lustre developed for HPC cluster environments (also tested with single
305 VMs). Lustre implementations offer improved performance and allow for multiple compute servers to connect to the Lustre host, where several servers are responsible for handling the transfer.

Cloud HPC Configuration Summary: We explored all the above options to have a complete set of solutions for different model cases and user needs that can be formulated around the cloud benchmark suite. In the standard CMAQ implementation, input
310 is read by all available cores while output is handled by only one of them. While the model performed as expected with single VMs, the code base had to be modified to correct issues with NFS-mounted storage in cluster environments that utilize more than ~180 cores. The code changes did not have an impact on the model results. If a parallel file system is present (i.e., Lustre, BeeGFS), users have the option to configure CMAQ with the parallel I/O algorithm (Parallel I/O implementation of CMAQ, 2023). Such implementations for CMAQ have been explored in previous versions of the model code base and performance
315 was investigated in more I/O demanding, higher spatially resolved simulations (Wong et al., 2015), but need to be thoroughly tested on the cloud with current compilers/hardware and were not considered at this stage of model benchmarking. It is, however, important to note that CMAQ input and output file sizes are highly dependent on the domain size and output file configuration options that can be simulation-specific, and users are encouraged to perform further analysis for their unique modeling application needs. Table 3 summarizes the different storage options that were included in the final set of benchmarks.
320 This list does not include certain storage solutions such as Azure NetApp Files (ANF), common internet file shares (CIFS), and the BeeGFS parallel file system, as these options were deemed either too expensive or created challenges when benchmarking CMAQ, e.g., CIFS does not allow for file links, ANF was more expensive for the CMAQ paradigm compared to other offerings from Microsoft, BeeGFS is not available as a service and needs additional server setup and tuning. The cluster configurations are described in Figures 5 and 6 as they demonstrate how the HPC resources on the Cloud are
325 dynamically provisioned by the ParallelCluster and CycleCloud User Interface. Figure 5 shows that the user logs into a head node on AWS and submits a CMAQ run using the SLURM scheduler. The HPC6a compute nodes are deployed only when CMAQ is running, with the number of nodes deployed specified by the SLURM commands within the run script. The other details of the configuration of the compute cluster including the type of head node, type of compute node, type of networking and storage available are specified in the yaml file that was used to create the cluster. As shown in Figure 6, the user also logs
330 into a head node on Microsoft Azure, with the number of compute nodes provisioned when the CMAQ job is running within the SLURM queue. The selection of head node, compute node, networking and storage type is made through a web interface to Azure CycleCloud UI.

340

Table 2: Overview of system configurations and technical capabilities for the two HPC systems that were used for benchmarking.

HPC Test System Description		
Cloud Service Provider	Microsoft Azure	Amazon Web Services
Service Name	CycleCloud	ParallelCluster
VM Name	Standard_HB120rs_v3	hpc6a.48xlarge
Processor	AMD EPYC 7V73X	AMD EPYC 7R13
CPU cores available	120	96
CPU speed (MHz)	1846	2650
Memory (GiB)	461	384
L3 Cache memory (MB)	96	192
Network Bandwidth (Gb/s)	200 (Nvidia HDR InfiniBand)	100 (Elastic Fabric Adapter -EFA)

345

Table 3: Overview of storage options for the two HPC systems that were used for benchmarking.

Storage Options		
Cloud Service Provider	Microsoft Azure	Amazon Web Services
Service Name	CycleCloud	ParallelCluster
Storage option 1 (/local)	Local NVMe SSDs in raid0 (2 * 960 GB NVMe - block)	N/A
Storage option 2 (/shared)	Built-in NFS: P30 Tier, Provisioned IOPS: 5000, Provisioned Throughput: 200 MB/s/TiB	Elastic Block Storage (EBS) – General purpose volumes (gp3): Provisioned IOPS: 3000, Provisioned Throughput: MB/s/TiB

Storage option 3 (*/data*)	NFS File share: Max IO/s 4024, Burst IO/s: 10000, Throughput rate: 203 MB/s	N/A
Storage option 4 (*/lustre*)	Lustre 150 – size 128 TB Performance profile: 150 MB/s/TiB Lustre 250 – size 128 TB Performance profile: 250 MB/s/TiB	Lustre SCRATCH_2 option: Size: 1 TB Network Throughput: 200 (1300 Burst) MB/s/TiB 240MB/s Disk Throughput: 200 MB/s/TiB (read), 100 MB/s/TiB (write)

350

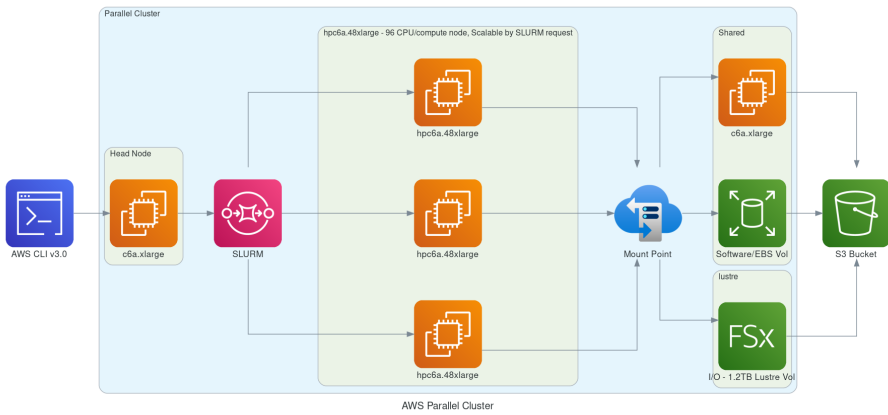



Figure 5: Schematic demonstrating AWS's ParallelCluster-based framework utilizing elastic compute nodes, scalable by SLURM request and different networked storage options (EBS, Lustre) with archival storage to AWS S3 Bucket.

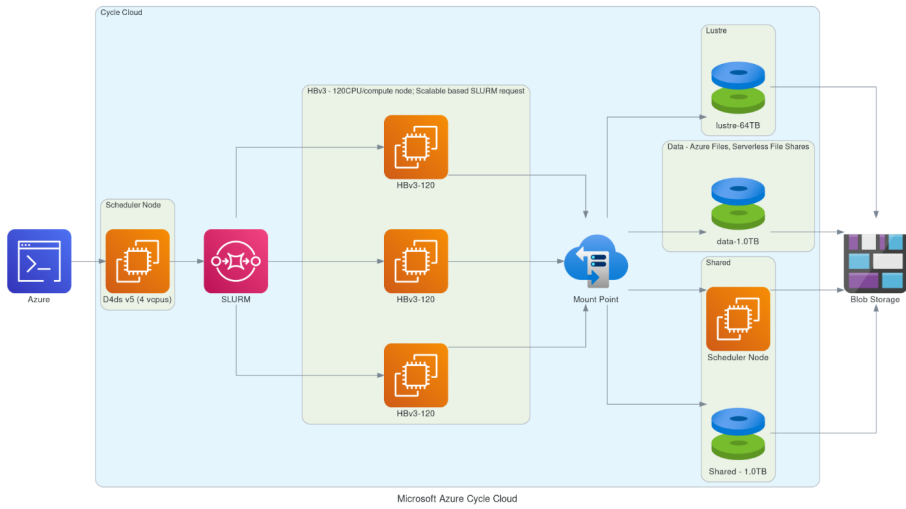



Figure 6: Schematic demonstrating Microsoft's Azure CycleCloud-based framework utilizing compute nodes scalable by SLURM request with different networked storage options (Lustre, Data, Shared) with archival storage to Azure Blob Storage.

4 Results

4.1 Single-node VM Timing Analysis

The CMAQ simulation will write two types of logfiles, a main logfile and processor-specific logfiles for each core/process. Model performance was evaluated using the main logfiles that include timings for the major science modules at each timestep: Vertical Diffusion (VDIFF), COUPLE (Converts units and couples or decouples concentration values with the density and Jacobian for transport), Horizontal Advection (HADV), Vertical Advection (ZADV), Horizontal Diffusion (HDIFF), DECOUPLE, Photolysis (PHOT), Cloud Process (CLDPROC), Chemistry (CHEM), and Aerosol (AERO). Horizontal advection is the most time consuming of the processes within CMAQ. This is most likely due to communication between processors during advection which requires information from neighboring cells to calculate advective fluxes. This is domain-dependent and there can be domains where the computational demand is very large (e.g., applications like Model Prediction Across Scales (MPAS)) (Gilliam et. al., 2021) that one may not see this trend till one uses thousands of

cores. In short, more cores results in less work per core but more time is needed for each core to communicate with each other. At the end of each simulation hour, species concentrations are output along with the timings printed for the output process (Data Output). It should be noted, this output process timing does not fully capture the total I/O time including initializing and shutting down the model (i.e., close all files, deallocate arrays). This unaccounted time component is derived from the difference between the total wall time (elapsed real time) and the sum of the sub-processes and was labeled as OTHER in the plots.

Model scalability is the measure of a system's ability to increase (or decrease) performance (and cost) in response to changes in system processing power, in our case determined by the specific resources (cores, memory, storage, and network protocols and bandwidth), and relies on MPI implementation and integration with the job manager (Slurm). Results from the benchmark case simulations performed in a single-node EPYC VM of Microsoft's Azure are presented in Figure 7. Figure 7 demonstrates good performance and efficiency scaling with both local and NFS Solid State Drive (SSD) storage options and some degree of a leveling off observed above 96 cores. As expected, the fastest local NVMe solution performs better than the same system with different storage options. Since NVMe is included in the default configuration, it is also the cheapest solution for a testing phase and despite its fixed volume it is sufficient for the benchmark domain and simple user needs (i.e., benchmarking, code development, testing). For larger domains and simulation periods, the SSD over NFS is a preferred solution that allows for larger volumes of data to be attached. Figure 8 provides a cloud benchmark case performance comparison broken down by model process component for each storage solution within Azure and AWS. A difference in the VM core allocation for hypervisor and background system tasks results in a different core count available for compute between the CSPs. For direct comparison with AWS, the system in both CSPs was optimized to utilize 96 out of the 120 available cores by employing process pinning, matching the region of best scalability observed at Figure 7. From Figure 8, it is evident that on Azure the Lustre filesystem competes very well with the local SSD solutions, followed by an additional performance difference for the NFS share (/data), and the proportionally slower but cheaper built-in NFS storage. On AWS, we could not provision a local SSD for the hpc6a.48xlarge single VM, so benchmark tests were limited to Lustre and EBS storage options. Results depicted in Figure 8 indicate that Lustre on AWS performed slightly slower compared to Azure, while the EBS option was a faster and more cost-effective solution for this CSP. However, users are advised to copy the input and output data in EBS (and local solution on Azure) as part of their workflow and additional time to complete the simulation should be accounted for.

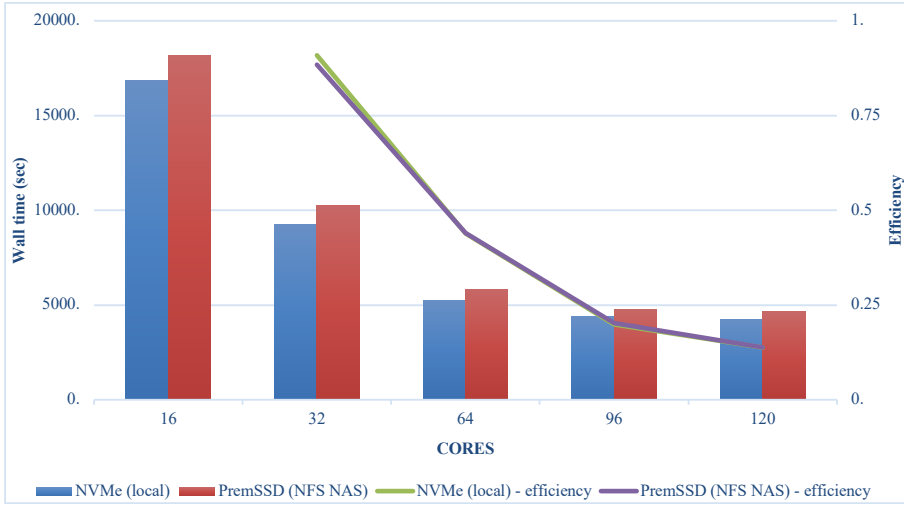



Figure 7: Performance comparison of the cloud benchmark suite (CBS_limited) simulations on a single VM of

415 **Microsoft's Azure utilizing 16 – 120 cores with a fast local SSD NVMe and premium SSD through the NFS client.**

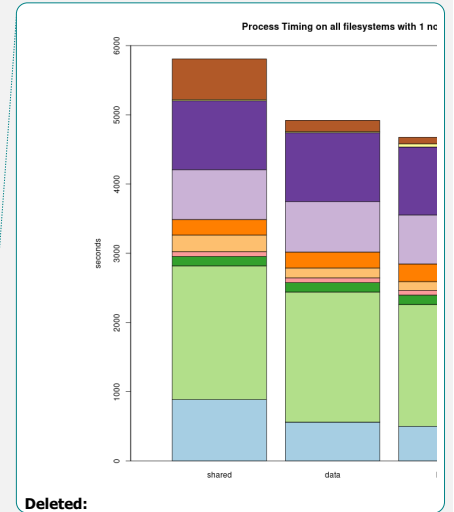
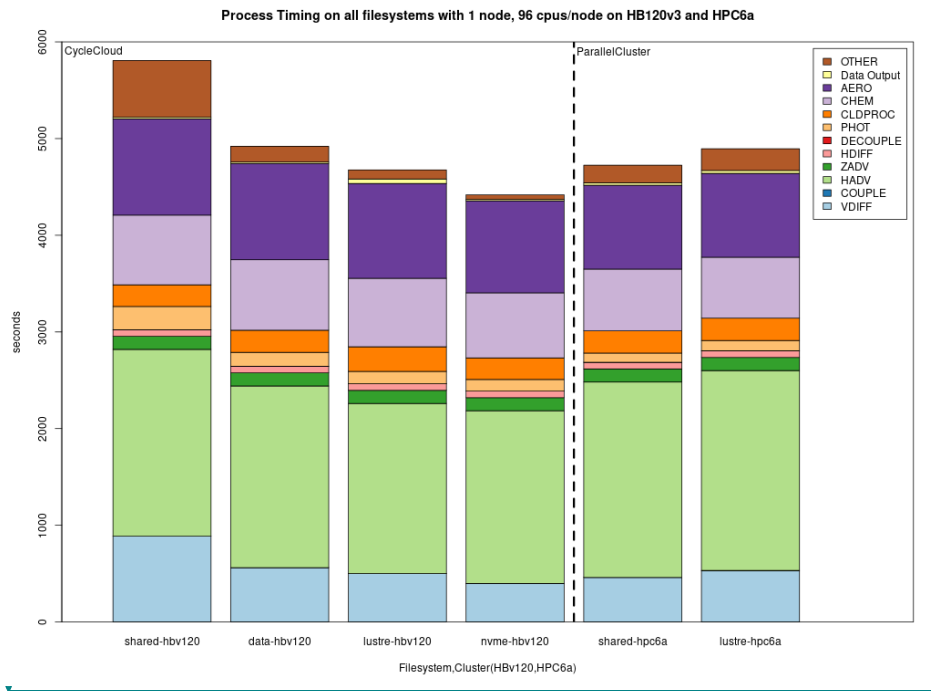



Figure 8: Performance comparison per model component for the cloud benchmark suite (CBS_limited) on a single node of Microsoft's Azure utilizing 96 cores with a a premium SSD through the NFS client (shared), a NetApp (ANF) solution (data), a Lustre filesystem, and a fast local SSD NVMe, and on a single node of AWS ParallelCluster utilizing 96 cores with the EBS (shared) and Lustre file system solutions (from left to right).

4.2 Optimization and Benchmarking on Multi-node Clusters

4.2.1 Process Pinning for L3 cache Optimization in the EPYC Processor Architectures

As mentioned before, in a managed job environment, AMD EPYC processors offer an option called 'process pinning' that can improve performance through more effective use of the L3 memory cache at the job submission level. This is another configuration option that should be evaluated, especially since implementations vary between CSPs and may change over time.

Figure 8 demonstrates the effect of process pinning on AWS and Azure, respectively. This option reduced simulation times on both systems using Lustre and shows that this effect can vary depending on the filesystem as well, with EBS volume use pointing to more substantial performance gains for AWS. Nevertheless, both Azure and AWS users should carefully consider such performance gains and further evaluate scaling under different pinning options for their domain and configuration.

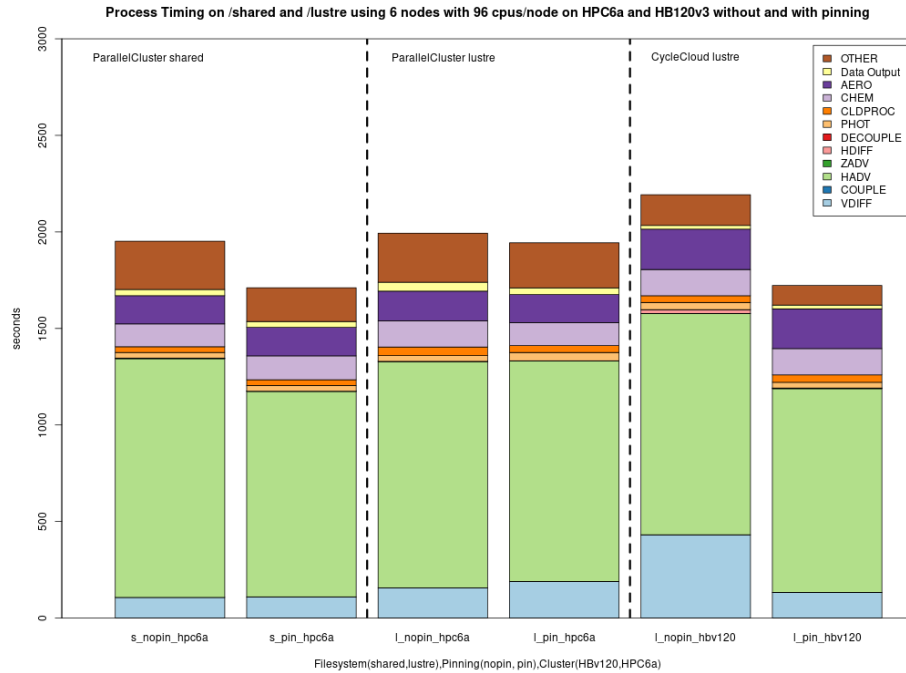



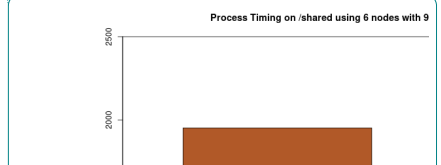
Figure 9: Effect of process pinning on shared and lustre and on AWS ParallelCluster (HPC6a) (576 cores) and on Azure CycleCloud HBv120 (576 cores) on lustre for cloud benchmark suite (CBS_limited).

4.2.2 Results on Multi-node Clusters with Different Storage Options Saving 12 Variables to 1-Layer CONC File

Figure 10 demonstrates the benchmark case results from simulations performed on Azure's CycleCloud clusters employing 1 – 6 nodes and two different Lustre implementations, a faster (250 MB/s/TiB) and a slower (150 MB/s/TiB) tier, both of size 100 TiB and for the NFS share and the slower built-in NFS solution, respectively. Depending on the end-user cost and overall

500 simulation needs, the slower solution can be a more cost-effective one, while the expensive option can be chosen when a faster turnaround time is necessary. In all cases, we observed diminishing performance gains when utilizing more than 2 nodes, with a plateau becoming apparent in the 3 – 6 node region. Figure 11 provides the performance breakdown for AWS's Lustre storage option and using the EBS (*shared*) storage option. On AWS ParallelCluster, a scratch Lustre option (200 MB/s/TiB) was

505 used. Lustre appears to be comparable in both CSP implementations, with minor differences that can be attributed to size of the filesystem that was provisioned (100 TiB on Azure Cycle Cloud, 1 TiB on AWS ParallelCluster), and the way the filesystem was parameterized (including stripe size) by different CSPs. EBS benchmarks were significantly faster, which makes it a potentially better alternative to Lustre for AWS instances. On AWS ParallelCluster, the Lustre filesystem is connected to the CMAS Center Open Data S3 bucket, and only the files that are used in the run script are copied from the S3 Bucket to Lustre. This strategy is used to identify resources as non-blocking (non-critical) and load these only when needed,

510 referred to as lazy loading. For the EBS benchmark, an AWS CLI script is used to copy the data from the S3 bucket to the EBS volume. The time taken to copy the data using the AWS CLI is higher (~15 min) than the time it takes for the data to be read from the S3 Bucket by Lustre (~300 seconds) and these timings were not included since the data were pre-loaded to the filesystems for these benchmarks.

Deleted: s

Deleted: 11 and 12

Deleted: Figures 13 and 14 show similar scaling results for the NFS share and the slower built-in NFS solution, respectively. …

Deleted: 1

Deleted: 5

Deleted: , while Figure 16 provides the equivalent

Deleted:

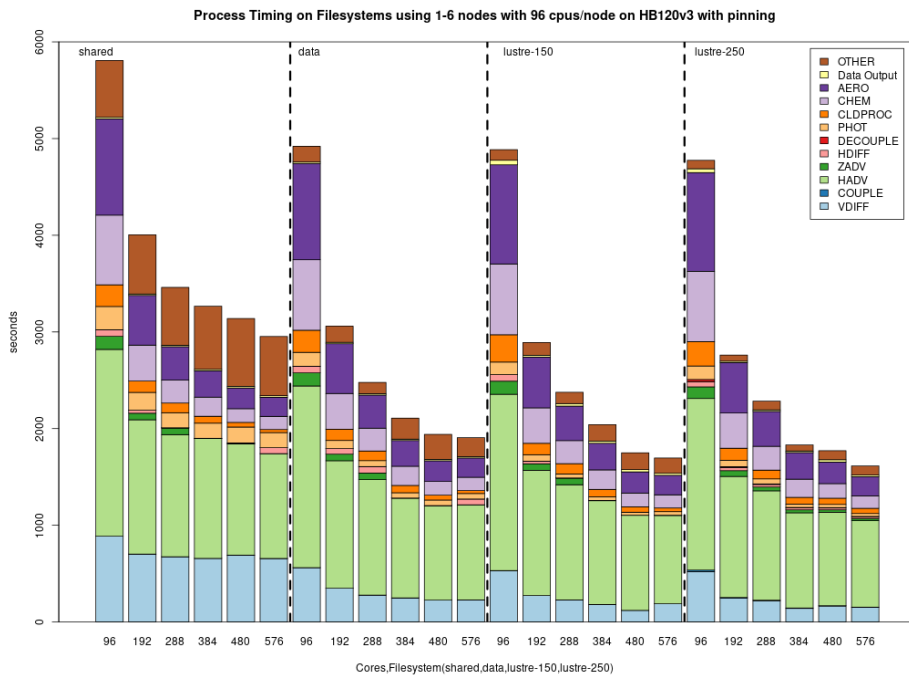



Figure 10: Performance of the cloud benchmark suite (CBS_limited) on Microsoft Azure CycleCloud environments using Filesystems (shared, data, Lustre 150 and Lustre 250), for I/O and code on *shared*.

615

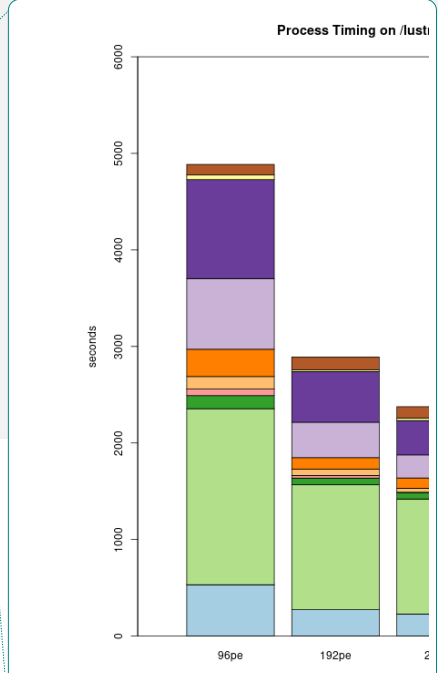
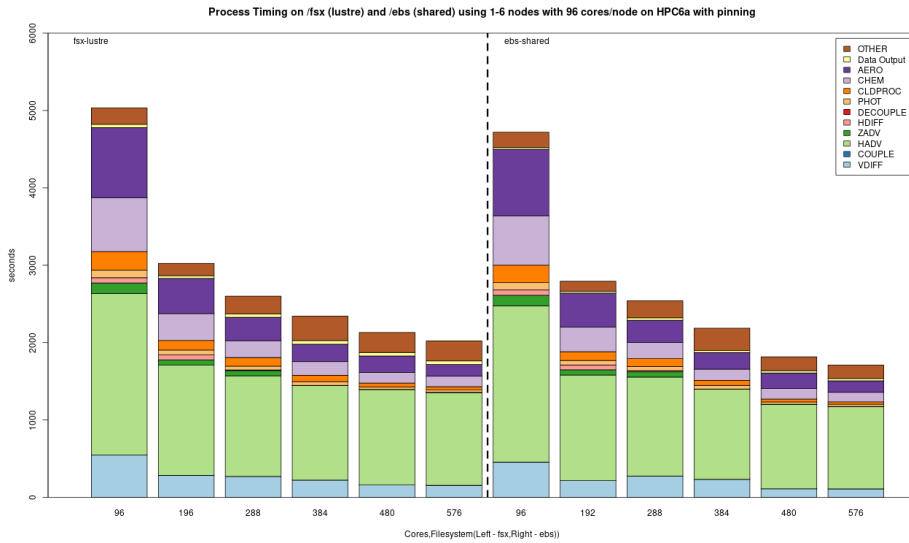



Figure 11: **Performance of the cloud benchmark suite (CBS_limited) on AWS ParallelCluster environments using /fsx (lustre) and /ebs (shared) filesystem.**

620 **4.2.3 Results on Multi-node Clusters with Different Storage Options saving 37 Variables to All-layer CONC File**

Figure 12 shows the performance of the EBS (shared) and the Lustre file systems using 96, 192 and 288 cores on AWS when either a limited number of output variables and one layer or the full number of output variables and all layers are saved to the 3D CONC file (creating and saving the largest output file possible under the cloud benchmark case – CBS_full).

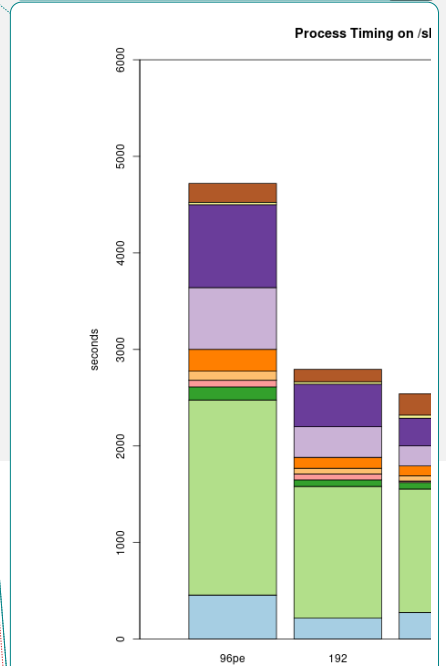
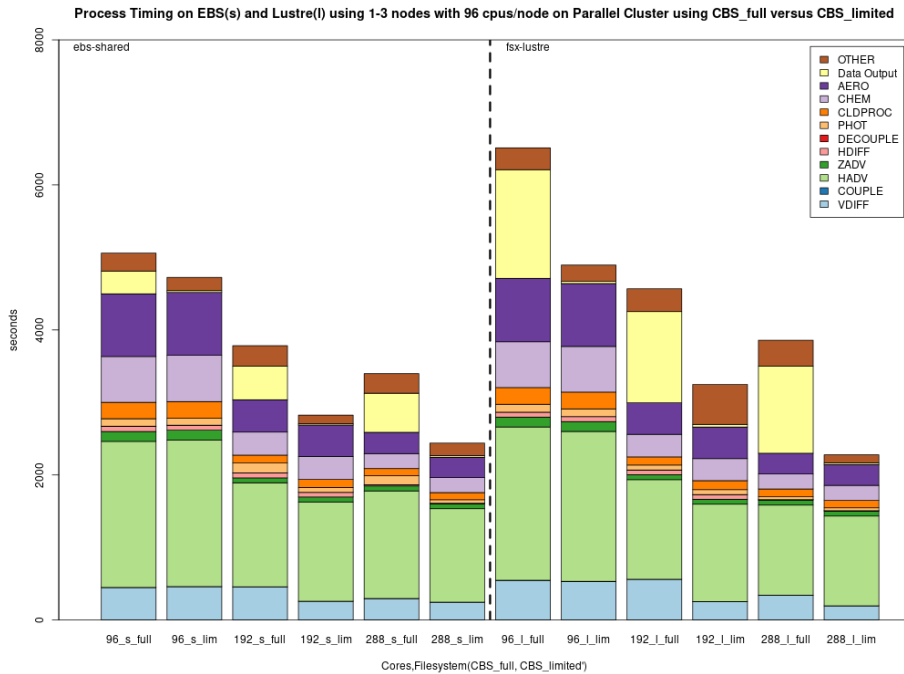



Figure 12: Performance of the 2-day cloud benchmark suite using 1-3 nodes with 96 cpus/node on AWS ParallelCluster environments using full output (CBS_full) versus limited output (CBS_limited) on EBS (shared(s)) and Lustre(l).

665

4.5 Cost Analysis of Compute Nodes

670 Table 4 shows comparison of the compute only costs associated with an annual simulation based on the cloud benchmark suite with limited output file options. The setup was based on 2-node cluster setups for both CSP's and the option of spot-pricing that was only available for Microsoft's Azure. It should be noted that spot instances can be preempted, resulting in a termination risk that the user should be aware of when designing their implementation. The compute node hpc6a.48xlarge is not provided as a spot instance, as the on-demand price is significantly discounted (60%). However, Amazon does offer spot prices for other

compute nodes. This analysis used on-demand pricing options to uniformly evaluate both systems. Users will need to implement code to check-point and recover from a simulation termination if they choose to use spot-pricing or be willing to restart simulations if spot instances are terminated.

725 **Table 4: Comparison of the compute costs for performing an annual simulation based on the cloud benchmark suite (CBS_limited) on 2-node clusters with on-demand and spot pricing tiers.** Note that these costs are indicative and do not include any other components of the cluster (i.e., storage, head node, etc.)

Compute Node	Cores	Nodes	Pricing	Cost per node ($)	CBS Wall Time (hour)	Extrapolated Annual Cost	Days to Complete Annual Simulation of CBS
HBv3-120	192	2	On-demand	3.6/hour	0.767	$1,007	5.83
HBv3-120	192	2	spot	1.4/hour	0.767	$392	5.83
hpc6a.48xl	192	2	On-demand	2.88/hour	0.839	$883	6.4

730

4.6 Results of using DESID module for emissions scaling sensitivity studies

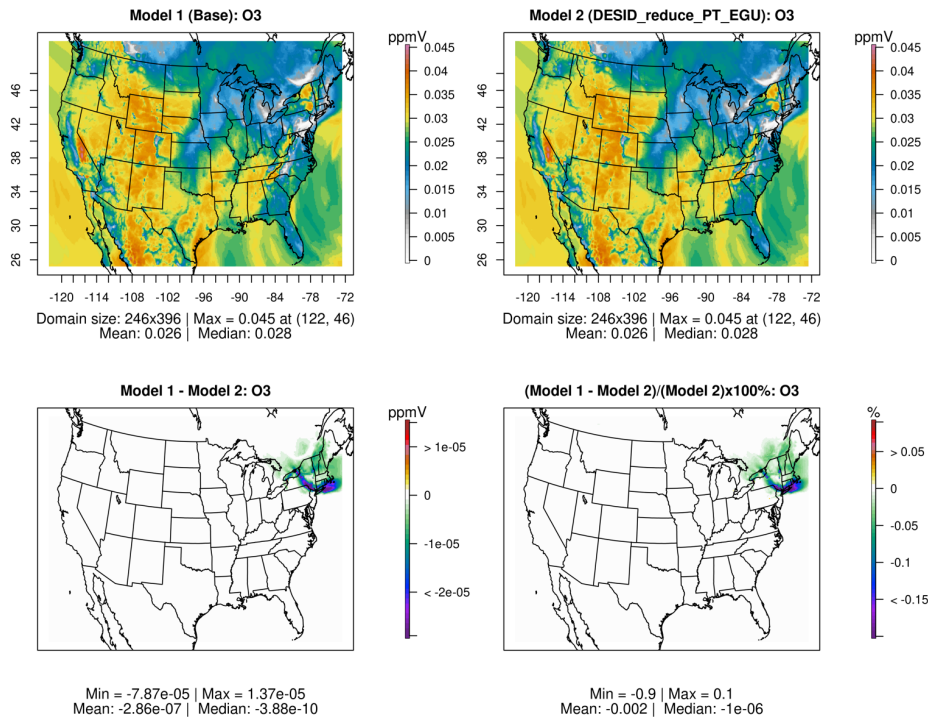
Online tutorials are available for CMAQv5.4 with instructions for running baseline and sensitivity examples using DESID.
735 Model to model comparison plots using spatial and time series analysis scripts in R and Python are also provided and demonstrated. Figure 13 shows the model to model mean spatial plots, difference plot, and percent difference plot for ozone from a base simulation and a sensitivity case where the DESID module was used to reduce the emissions from power plants in New York State.

740

Figure 13: Model-to-model comparison for a base case of CMAQv5.3.3, and a sensitivity case where DESID was used to reduce the point source electric generating unit (PT_EGU) emissions in NY State by 25%.

5 Discussion

5.1 Benefits of Proposed Cloud-based Implementation

Previous efforts of bringing CMAQ to the cloud demonstrated the potential of packaging the model along with other components as a standalone service optimized for small benchmark domains and low-cost VMs (Zhang et al., 2019). Currently, running complex, computationally demanding models on the cloud presents new options for optimizing workflow, performance, and costs with access to HPC resources. A major implication is a fast deployment of such infrastructures with precompiled software snapshots and preloaded data that are easy to configure and customize according to the user needs. This work evaluates how to run CMAQ on two CSPs using their cluster management tools (ParallelCluster and CycleCloud) and illustrates several issues that should be considered for building HPC clusters on the cloud. We observed that, despite their differences, both AWS and Azure performed similarly, and had comparable performance to onsite HPC implementations used in earlier phases of this work. The online tutorials provide guidance for selecting CSP cost-effective options (compute nodes, storage, and high performance networking) for the current CONUS benchmark suite and can be used as a guide for benchmarking more demanding CMAQ applications, such as the coupled model (WRF-CMAQ) (Wong et. al 2012), the Integrated Source Apportionment Method (CMAQ-ISAM) (Cohen and Napelenok, 2011), or simulations with higher horizontal or vertical grid resolution.

5.2 Impact of Storage Options and process pinning on CSPs

Choice of a storage option is shown to have an important impact on simulation run times. Figure 7 shows best performance on NVMe drives, which are only available on single virtual machines. To fully utilize the potential of HPC solutions, the Lustre storage option is advisable for the Azure ecosystem. For AWS, EBS offers a cost-effective alternative to Lustre. However additional data transfers may be required (e.g., copying input and output data to/from the S3 Bucket) in the workflow if the ParallelCluster is configured to have the EBS volumes deleted when the cluster is terminated. Scaling performance was improved when both the code and the data files reside on /lustre and may also improve if both the code and the I/O is on local storage (/nvme) or Azure File Share (/data). The effect of process pinning resulted in modestly improved timings on both EBS and Lustre on AWS ParallelCluster. Process pinning resulted in significant timing improvements for Lustre on Azure CycleCloud. Process pinning resulted in improved performance on /shared, /data and /lustre using Azure's CycleCloud, with the best performance being on Lustre.

5.3 Scalability

A key issue that is brought up in the results section is the model scalability which exhibits a diminishing return as the systems are scaled out across more cores and nodes. In general, scalability depends on the domain characteristics (domain size, resolution) and the hardware. Domain decomposition can significantly reduce performance when domain is highly decomposed, i.e., only a few grid cells are assigned to each core. While our results focus on a fixed size Cloud Benchmark

Suite, we expect improved scalability with a higher-resolution domain, as it would increase the workload per core. Compared to typical WRF benchmarks where a 2.5-km x 2.5-km resolution CONUS domain (63 million grid cells) is considered a typical case that scales well up to a few hundred cores, the 12-km x 12-km CMAQ CBS (97 thousand grid cells) is comparatively too small of a problem to scale similarly.

5.4 Future Research Recommendations

The online tutorials and documentation include recommendations for future work such as: 1) running the benchmark using new releases of virtual machines (Elastic Compute Cloud (EC2) Instances or Microsoft Azure VMs), 2) building with the EPYC processor including Standard_HB176rs_v4 on Azure Cycle Cloud and on new releases of the Arm-based AWS Graviton-3 processor using c7g.16xlarge on ParallelCluster, 3) running the benchmark using a parallel I/O API implementation and other efficiency improvements in the source code that can be compiler and processor architecture specific. Impact on performance needs to be examined after each model release and for each model configuration and input platform data which varies by year and model parameterization.

5.5 Using Cloud Service Provider versus On-Premise Compute Servers

The choice of conducting simulations on CSPs versus an on-premise option comes down to the cost and resources of the on-premise option, and the specifics of the work to be accomplished. In some cases, organizational investments to support computational and data needs may effectively subsidize the cost of an on-premise solution. Even when an on-premise option is available, it may be reasonable to utilize both on-premise and CSP services to take advantage of the flexibility and scalability of building custom infrastructure and workflows within a CSP. Our cost estimates for the Cloud Benchmark Suite support the conclusion by Zhuang et al. (2020) that atmospheric modeling in the cloud can be a cost competitive alternative to more traditional HPC systems.

6. Conclusions

This work provides reproducible workflows to facilitate provisioning of HPC clusters on the cloud, setting up and running CMAQ and using performance analysis tools and profilers to optimize the HPC systems available from CSPs. The very nature of cloud implementations comes with the advantage that CSPs are continuously offering advancements in compute, memory, and storage resources. Different CSPs use different hardware versions (EPYC processor versions), with different hypervisor on a different Nitro chip, and SLURM implementations, that result in different number of cores available per node. Azure provided manual pinning instructions that required a detailed understanding of the CPU architecture and SLURM scheduler.

In the case of AWS, the implementation resulted in fewer cores/node available to the user (96 vs. 128), but automatically bound the specific process to a core and did not require additional fine-tuning steps (manual pinning had less impact on timings). CMAQ is also continuously under development and while the method presented here used CMAQv5.3.3 with the 12US2 benchmark and the CB6 mechanism, it may be extended to compile and run new versions of CMAQ, new mechanisms such as the Community Regional Atmospheric Chemistry Multiphase Mechanism (CRACMM) (Pye et. al., 2023), and other configurations including WRF-CMAQ and CMAQ-ISAM. Our HPC in the cloud paradigm allows researchers to improve their workflow and access a menu of specialized HPC resources offered by cloud computing vendors resulting in faster time to solution. These tutorials by the CMAS community are designed to facilitate use of best practices for HPC cloud provisioning, increase cross-institution collaborations, and improve efficiency in code development and deployment cycles.

Code and data availability

The code for CMAQ is available and referenced here:

https://github.com/USEPA/CMAQ/

https://doi.org/10.5281/zenodo.5213949

The code for the AWS cloud implementation is available and referenced here:

https://github.com/CMASCenter/pcluster-cmaq/tree/CMAQv5.3.3

https://doi.org/10.5281/zenodo.10696908

The code for the Azure cloud implementation is available and referenced here:

https://github.com/CMASCenter/cyclecloud-cmaq/tree/CMAQv5.3.3

https://doi.org/10.5281/zenodo.10696804

Data inputs for the Benchmark Suite are available and referenced here:

https://registry.opendata.aws/cmas-data-warehouse/

https://doi.org/10.15139/S3/CFU9UL

Tutorials with instructions on running CMAQ version 5.3.3 and above on the cloud are available through:

https://cyclecloud-cmaq.readthedocs.io/en/cmaqv5.3.3/

https://pcluster-cmaq.readthedocs.io/en/cmaqv5.3.3/

Supplement

The supplement related to this article is available online as an Appendix below.

Author contributions

CE and LA share joint first authorship of the study. CE and LA designed the study, implemented the parameterizations and setups, performed the simulations, analyzed the results, and drafted the original draft of the paper. CC and DW contributed to various model optimizations and troubleshooting model performance. RZ, MR and JM contributed to provisioning Microsoft

Azure resources for the modeling, troubleshooting modeling issues on Azure and comparisons against AWS. KF, FS and SF

960 contributed to refinement of model configurations, troubleshooting and interpretation of results. SA conceived the study and provided project oversight and guidance. All authors contributed to comments and revisions of the original draft.

Acknowledgements

We acknowledge Amazons' Open Data Program for hosting the datasets and Microsoft for providing cloud computing credits

965 for the work performed in this study. Steve Roach of Microsoft provided guidance on configuring the CycleCloud, using pinning to improve performance with the HB120v3 compute nodes, and access to the Lustre filesystem. Rafa Salas of Microsoft provided access to cloud experts and facilitated a grant to UNC with computing credits. Timothy Brown and Tommy Johnston of AWS provided guidance on configuring ParallelCluster using HPC6a compute nodes. Chris Stoner of AWS provided guidance on enrollment in the AWS Open Data Sponsorship Program and technical assistance with the Handbook

970 for Data Providers that was used to create the CMAS Data Warehouse, and further making the datasets searchable on the AWS Data Exchange. We also acknowledge the contributions of Zac Adelman (LADCO), Steve Fine and Fahim Sidi (EPA) as co-leaders of the Air Quality Modeling in the Cloud Computing Workgroup, who helped guide our modeling effort on the cloud.

975 **Disclaimer.**

This paper has been subjected to an EPA review and was approved for publication. The views expressed here are those of the authors and do not necessarily reflect the views and policies of the US Environmental Protection Agency (EPA), or any of the CSPs named here.

980 **Financial support**

The U.S. EPA, through its Office of Research and Development, partially funded and collaborated in the research described here under EP-W-16-014 to UNC Chapel Hill. Microsoft provided credits for performing the necessary benchmark simulations on Azure.

985

Competing interests

UNC through its Information Technology Services Research Computing division received cloud computing credits from Microsoft that were used for the work performed here, and received sponsorship from AWS for enrollment in the Open Data Program that are used to host the datasets mentioned here.

990

Appendix

1. HPC Architectures and CSP Services

A growing number of companies ranging from large enterprises such as Amazon, Microsoft, and Google, to a spectrum of cloud-focused computer firms, have a strong presence with evolving portfolios in what is broadly defined as "public cloud infrastructure". Typically, cloud computing is provided through at least four types of services summarized in Figure A1: Infrastructure as a service (IaaS), Platform as a Service (PaaS), Software as a service (SaaS), and Data as a Service (DaaS) (Mell and Grance, 2011; Chang et al., 2010; Yuan, 2016). IaaS products (Amazon Web Services, Microsoft Azure, Google Cloud, etc.) allow organizations and end-users to manage their system resources (i.e., servers, network, data storage) on the cloud. PaaS products (Windows Azure, Google app engine, etc.) allow businesses and developers to host, build, and deploy consumer-facing apps. The most important contrast between IaaS and PaaS is that IaaS gives users more administrative control over operating systems and system resources, while PaaS gives consumers the ease of use of provided applications, but limits access to choices about the operating system and system resources. SaaS products are among the most popular cloud computing services (Microsoft 365, Google docs, Adobe cloud, etc.) offering out-of-the-box, simple solutions that usually target common end users and operate under a subscription model. DaaS, the least well-defined type of service, describes cloud-based software tools used for working with data, such as managing data in a data warehouse entity, processing, or analyzing with software tools, typically enabled by SaaS technologies under a subscription model. HPC applications require administrative access to networking, hardware, and storage configurations, and therefore need infrastructure as a service (IaaS) and infrastructure as code (IaaC) level of control that are provided by the ParallelCluster and CycleCloud cluster management services.

Figure A1: Architecture of on-site and cloud-based services demonstrating the degree of end-user control.

020

2. Tools for Storage Performance Monitoring

2a. Amazon CloudWatch

Amazon CloudWatch (Amazon CloudWatch, 2024) and Azure Monitor™ (Azure Monitor, 2024) allow you to monitor the I/O throughput of a filesystem while running an application in real time. Amazon CloudWatch output shown in Figure A2 can be used to further investigate the impact of storage on performance seen when writing additional model outputs (e.g., full layered 3-D instantaneous model concentrations) to different storage options. While AWS CloudWatch shows higher throughput on Lustre than shared, the benchmark performance is faster on shared than Lustre. This may be due to larger disk caches or faster latencies on the EBS volume. The Lustre performance may be improved by using a persistent volume versus scratch volume that was used in this study.

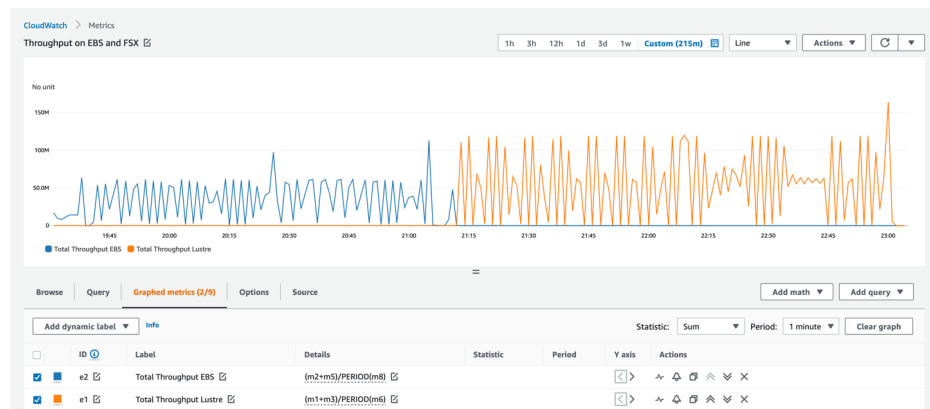



Figure A2. Amazon™ CloudWatch™ throughput measurement of the first day of the cloud benchmark suite using 96 cores on ParallelCluster using different filesystems, EBS (blue) and Lustre (orange) for full output (all variables, all layers) in CONC file (CBS_full).

2b. Azure Monitor

1040   Figure A3 displays the read and write throughput, and client latency metrics from the Azure Monitor for the cloud benchmark suite using Azure Managed Lustre (250 MB/s). Insights into the I/O performance are facilitated by the web tools available from each CSP.

1045

Figure A3. Azure Monitor™ Cloud Metric latency and throughput measurement of data read, and data write for the cloud benchmark suite using 96 cores on CycleCloud using the Lustre filesystem for full output (all variables, all layers) in CONC file for the cloud benchmark suite (CBS_full).

1050

2c ARM MAP™ Code Profiler Analysis for Azure CycleCloud and AWS ParallelCluster

  As mentioned before, examining the main log and associated timing plots does not precisely capture the time spent in I/O tasks. The use of a code profiler such as the ARM MAP™ Profiler (ARM Ltd., 2022) can provide better insights on the model components, along with more detailed capture of the I/O tasks by different code routines. Results from applying the ARM 1055 MAP™ code profiler on a single day benchmark simulation using the limited I/O benchmark for each CSP cluster and storage offering are presented in Figure A4. We can clearly see a very similar behavior in both systems with the model using a Lustre

filesystem. For */shared*, AWS's EBS solution is performing much better, with less time spent in I/O that allows speeds closer to physical, "bare-metal" server equivalents. For Azure, the */shared* volume has proportionately higher time spent in I/O percentage and lower percentage spent on computation. In earlier tests without a profiler, the Azure NetApp Filesystem (ANF) solution provided better I/O performance but due to considerably higher cost due to a minimum file size of 4TB we chose not to include an ANF setup in the profiler tests. The performance improvements going from */shared* to */data* and Lustre on Azure's CycleCloud are also demonstrated by comparing Figures A5, A6, A7, where orange at the beginning of the run indicates the I/O portions that get reduced as we utilize faster storage options.

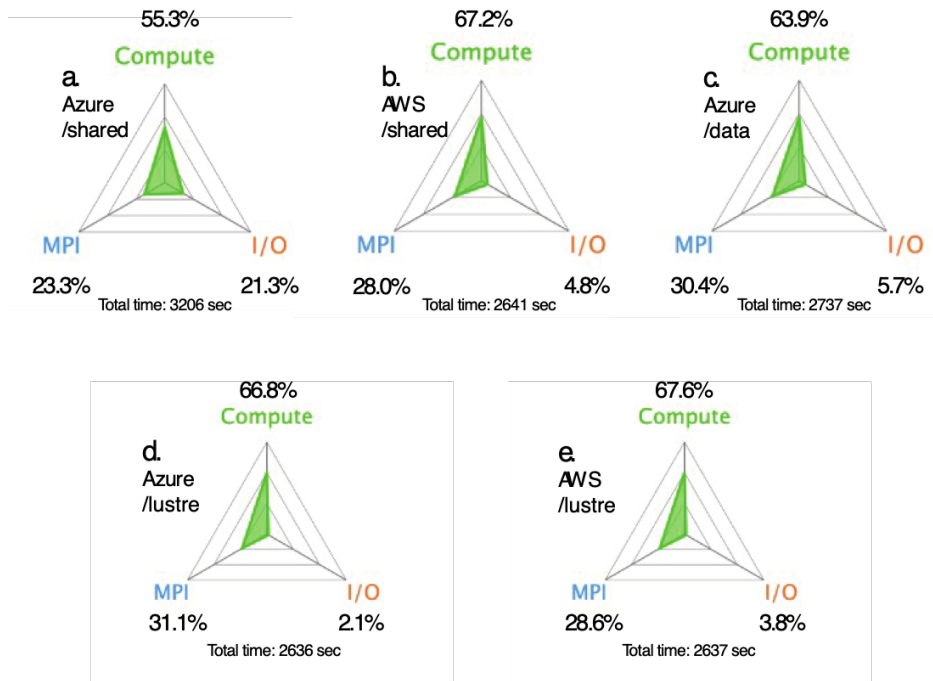
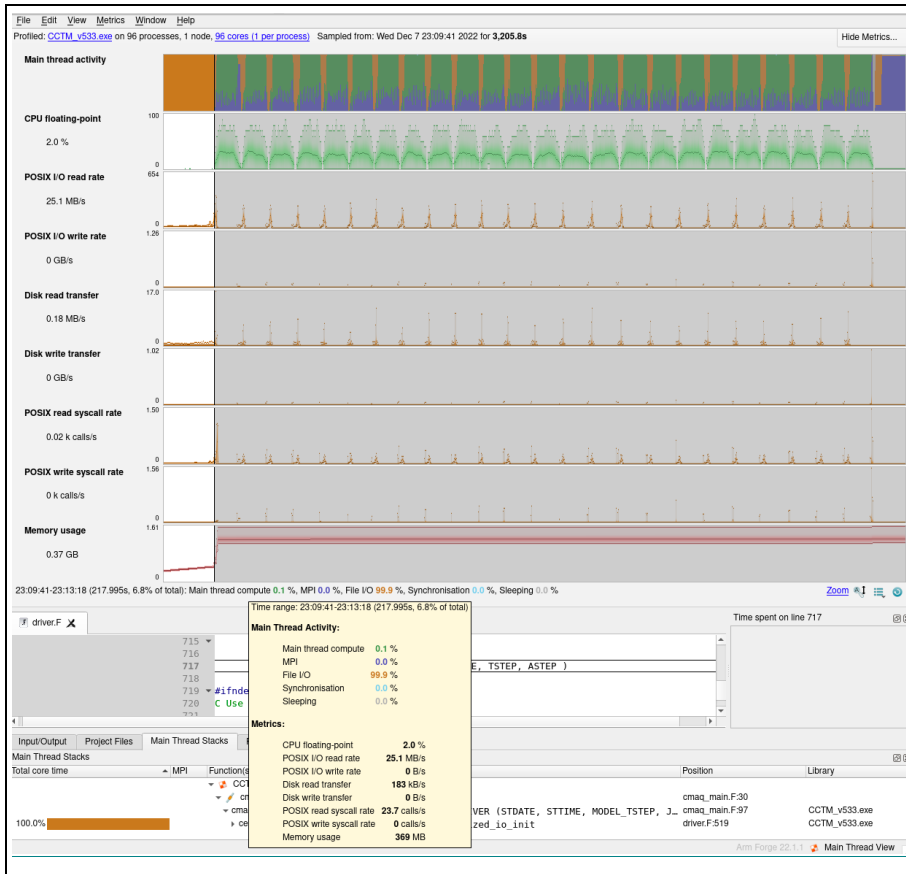



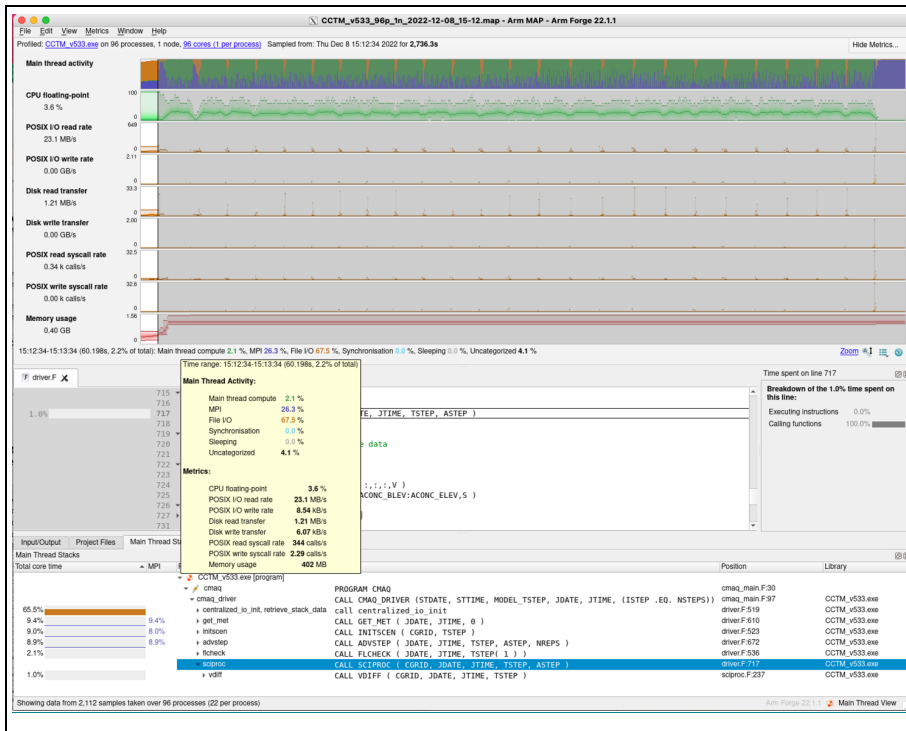
Figure A5: ARM MAP™ Code profiler single day performance analysis for Azure's CycleCloud using the NFS /shared filesystem.

075 **Figure A6: ARM MAP™ Code profiler single day performance analysis for Azure's CycleCloud using the /data filesystem.**

Figure A7: ARM MAP™ Code profiler single day performance analysis for Azure's CycleCloud using the Azure Managed *Lustre* Filesystem.

References

Amazon CloudWatch™, https://aws.amazon.com/cloudwatch/, last access: March 22, 2024.

Appel, K. W., Napelenok, S. L., Foley, K. M., Pye, H. O. T., Hogrefe, C., Luecken, D. J., Bash, J. O., Roselle, S. J., Pleim, J. E., Foroutan, H., Hutzell, W. T., Pouliot, G. A., Sarwar, G., Fahey, K. M., Gantt, B., Gilliam, R. C., Heath, N. K., Kang, D., Mathur, R., Schwede, D. B., Spero, T. L., Wong, D. C., and Young, J. O.: Description and evaluation of the Community

Multiscale Air Quality (CMAQ) modeling system version 5.1, Geosci Model Dev, 10, 1703–1732, https://doi.org/10.5194/gmd-10-1703-2017, 2017.

1100 Appel, K. W., Bash, J. O., Fahey, K. M., Foley, K. M., Gilliam, R. C., Hogrefe, C., Hutzell, W. T., Kang, D., Mathur, R., Murphy, B. N., Napelenok, S. L., Nolte, C. G., Pleim, J. E., Pouliot, G. A., Pye, H. O. T., Ran, L., Roselle, S. J., Sarwar, G., Schwede, D. B., Sidi, F. I., Spero, T. L., and Wong, D. C.: The Community Multiscale Air Quality (CMAQ) model versions 5.3 and 5.3.1: system updates and evaluation, Geosci Model Dev, 14, 2867–2897, https://doi.org/10.5194/gmd-14-2867-2021, 2021.

105 Arm Ltd. <2022>. Arm MAP. https://www.linaroforge.com/

AWS, The security design of AWS Nitro system: https://docs.aws.amazon.com/pdfs/whitepapers/latest/security-design-of-aws-nitro-system/security-design-of-aws-nitro-system.pdf, last access: 21 February 2023.

Azure Monitor™, https://learn.microsoft.com/en-us/azure/azure-monitor/overview, last access: March 22, 2024

Byun, D. and Schere, K. L.: Review of the Governing Equations, Computational Algorithms, and Other Components of the

1110 Models-3 Community Multiscale Air Quality (CMAQ) Modeling System, Appl Mech Rev, 59, 51–77, https://doi.org/10.1115/1.2128636, 2006.

Campbell, P. C., Jiang, W. (Rick), Moon, Z., Zinn, S., and Tang, Y.: NOAA's Global Forecast System Data in the Cloud for Community Air Quality Modeling, Atmosphere, 14, 1110, https://doi.org/10.3390/atmos14071110, 2023.

Chang, W. Y., Abu-Amara, H., and Sanford, J. F.: Transforming Enterprise Cloud Services, 133–187,

1115 https://doi.org/10.1007/978-90-481-9846-7_4, 2010.

Chen, X., Huang, X., Jiao, C., Flanner, M. G., Raeker, T., and Palen, B.: Running climate model on a commercial cloud computing environment: A case study using Community Earth System Model (CESM) on Amazon AWS, Comput Geosci, 98, 21–25, https://doi.org/10.1016/j.cageo.2016.09.014, 2017.

Chui, T. C. Y., Siuta, D., West, G., Modzelewski, H., Schigas, R., and Stull, R.: On producing reliable and affordable

1120 numerical weather forecasts on public cloud-computing infrastructure, J Atmos Ocean Tech, 36, 491–509, https://doi.org/10.1175/jtech-d-18-0142.1, 2019.

CMAS's Data Warehouse on AWS: https://registry.opendata.aws/cmas-data-warehouse/, last access: 21 February 2023.

The BAMS/EDSS/Models-3 I/O API: User Manual, Related Programs, and Examples: https://www.cmascenter.org/ioapi/documentation/all_versions/html/, last access: 21 February 2023.

125 Cohan, D., Napelenok, S. (2011). Air Quality Response Modeling for Decision Support. Atmosphere, 2(3): 2030407. doi: 10.3390/atmos2030407

Eastham, S. D., Long, M. S., Keller, C. A., Lundgren, E., Yantosca, R. M., Zhuang, J., Li, C., Lee, C. J., Yannetti, M., Auer, B. M., Clune, T. L., Kouatchou, J., Putman, W. M., Thompson, M. A., Trayanov, A. L., Molod, A. M., Martin, R. V., and Jacob, D. J.: GEOS-Chem High Performance (GCHP v11-02c): a next-generation implementation of the GEOS-Chem

1130 chemical transport model for massively parallel applications, Geosci Model Dev, 11, 2941–2953, https://doi.org/10.5194/gmd-11-2941-2018, 2018.

Parallel I/O implementation of CMAQ: https://github.com/USEPA/CMAQ/blob/main/DOCS/Users_Guide/Appendix/CMAQ_UG_appendixD_parallel_implementation.md#d3-parallel-io., last access: 21 February 2023.

135 U.S. EPA: The Community Multiscale Air Quality (CMAQ) model, https://zenodo.org/records/5213949, 2023.

Foley, K. M., Roselle, S. J., Appel, K. W., Bhave, P. V., Pleim, J. E., Otte, T. L., Mathur, R., Sarwar, G., Young, J. O., Gilliam, R. C., Nolte, C. G., Kelly, J. T., Gilliland, A. B., and Bash, J. O.: Incremental testing of the Community Multiscale Air Quality (CMAQ) modeling system version 4.7, Geosci Model Dev, 3, 205–226, https://doi.org/10.5194/gmd-3-205-2010, 2010.

140 GEOS-Chem registry of open data: https://registry.opendata.aws/geoschem-input-data/, last access: 21 February 2023.

Ghatrehsamani, D., Denninnart, C., Bacik, J., and Salehi, M. A.: The Art of CPU-Pinning: Evaluating and Improving the Performance of Virtualization and Containerization Platforms, 49th Int Conf Parallel Process - Icpp, 1–11, https://doi.org/10.1145/3404397.3404442, 2020.

Gilliam, R. C., J. A. Herwehe, O. R. Bullock, Jr., J. E. Pleim, L. Ran, P. C. Campbell, and H. Foroutan, 2021. Establishing
145 the suitability of the Model for Prediction Across Scales for global retrospective air quality modeling. J. Geophys. Res.: Atmospheres, 126, e2020JD033588. https://doi.org/10.1029/2020JD033588

Mell, P. and Grance, T.: The NIST Definition of Cloud Computing, Special Publication (NIST SP), National Institute of Standards and Technology, Gaithersburg, MD, 2011.

Murphy, B. N., Nolte, C. G., Sidi, F., Bash, J. O., Appel, K. W., Jang, C., Kang, D., Kelly, J., Mathur, R., Napelenok, S.,
150 Pouliot, G., and Pye, H. O. T.: The Detailed Emissions Scaling, Isolation, and Diagnostic (DESID) module in the Community Multiscale Air Quality (CMAQ) modeling system version 5.3.2, Geosci. Model Dev., 14, 3407–3420, https://doi.org/10.5194/gmd-14-3407-2021, 2021.

NOAA's Big Data Program: https://ncics.org/data/noaa-big-data-project/, last access: 21 February 2023.

Otte, T. L. and Pleim, J. E. The Meteorology-Chemistry Interface Processor (MCIP) for the CMAQ modeling system:
155 updates through MCIPv3.4.1. Geosci. Model Dev. 3, 243–256, https://doi.org/10.5194/gmd-3-243-2010, 2010.

Podzimek, A., Bulej, L., Chen, L. Y., Binder, W., and Tuma, P.: Analyzing the Impact of CPU Pinning and Partial CPU Loads on Performance and Energy Efficiency, Ieee Acm Int Symp, 1–10, https://doi.org/10.1109/ccgrid.2015.164, 2015.

Powers, J. G., Werner, K. K., Gill, D. O., Lin, Y.-L., and Schumacher, R. S.: Cloud Computing Efforts for the Weather Research and Forecasting Model, B Am Meteorol Soc, 102, E1261–E1274, https://doi.org/10.1175/bams-d-20-0219.1, 2021.

160 Pye, H. O. T., Place, B. K., Murphy, B. N., Seltzer, K. M., D'Ambro, E. L., Allen, C., Piletic, I. R., Farrell, S., Schwantes, R. H., Coggon, M. M., Saunders, E., Xu, L., Sarwar, G., Hutzell, W. T., Foley, K. M., Pouliot, G., Bash, J., and Stockwell, W. R.: Linking gas, particulate, and toxic endpoints to air emissions in the Community Regional Atmospheric Chemistry Multiphase Mechanism (CRACMM), Atmos. Chem. Phys., 23, 5043–5099, https://doi.org/10.5194/acp-23-5043-2023, 2023.

Simonson, A., Brown, O., Dissen, J., Kearns, E. J., Szura, K., and Brannock, J.: Big Data Analytics in Earth, Atmospheric,
165 and Ocean Sciences, Special Publ, 65–94, https://doi.org/10.1002/9781119467557.ch4, 2022.

Skamarock, W. C., Klemp, J. B., Dudhia, J., Gill, D. O., Liu, Z., Berner, J., Wang, W., Powers, J. G., Duda, M. G., Barker, D., and Huang, X.: A Description of the Advanced Research WRF Model Version 4.3, No. NCAR/TN-556+STR, https://doi.org/http://dx.doi.org/10.5065/1dfh-6p97, 2021.

1170   Unidata. Network Common Data Form (NetCDF). https://doi.org/10.5065/D6H70CW6, last access: 14 October 2023.

Wong, D. C., Yang, C. E., Fu, J. S., Wong, K., and Gao, Y.: An approach to enhance pnetCDF performance in environmental modeling applications, Geosci Model Dev, 8, 1033–1046, https://doi.org/10.5194/gmd-8-1033-2015, 2015.

Wong, D.C., Pleim, J., Mathur, R., Binkowski, F., Otte, T., Gilliam, R., Pouliot, G., Xiu, A., and Kang, D. (2012). WRF-CMAQ two-way coupled system with aerosol feedback: software development and preliminary results. *Geosci. Model Dev.*,

1175   5, 299-312. doi: 10.5194/gmd-5-299-2012

Yuan, M.: Cloud Computing in Ocean and Atmospheric Sciences, 385–391, https://doi.org/10.1016/b978-0-12-803192-6.00020-7, 2016.

Zhang, C., Di, L., Sun, Z., Lin, L., Yu, E. G., and Gaigalas, J.: Exploring cloud-based Web Processing Service: A case study on the implementation of CMAQ as a Service, Environ Modell Softw, 113, 29–41,

1180   https://doi.org/10.1016/j.envsoft.2018.11.019, 2019.

Zhuang, J., Jacob, D. J., Lin, H., Lundgren, E. W., Yantosca, R. M., Gaya, J. F., Sulprizio, M. P., and Eastham, S. D.: Enabling High-Performance Cloud Computing for Earth Science Modeling on Over a Thousand Cores: Application to the GEOS-Chem Atmospheric Chemistry Model, J Adv Model Earth Sy, 12, https://doi.org/10.1029/2020ms002064, 2020.

1185

Deleted: ¶

Page 19: [1] Deleted	**Adams, Liz**	**6/3/24 5:27:00 PM**
Page 20: [2] Deleted	**Adams, Liz**	**6/18/24 5:17:00 PM**
Page 20: [2] Deleted	**Adams, Liz**	**6/18/24 5:17:00 PM**
Page 20: [2] Deleted	**Adams, Liz**	**6/18/24 5:17:00 PM**
Page 20: [2] Deleted	**Adams, Liz**	**6/18/24 5:17:00 PM**
Page 20: [2] Deleted	**Adams, Liz**	**6/18/24 5:17:00 PM**
Page 20: [2] Deleted	**Adams, Liz**	**6/18/24 5:17:00 PM**
Page 20: [2] Deleted	**Adams, Liz**	**6/18/24 5:17:00 PM**
Page 20: [3] Deleted	**Adams, Liz**	**6/3/24 5:27:00 PM**
Page 20: [3] Deleted	**Adams, Liz**	**6/3/24 5:27:00 PM**
Page 20: [4] Deleted	**Adams, Liz**	**6/3/24 5:29:00 PM**
Page 22: [5] Deleted	**Adams, Liz**	**6/3/24 5:34:00 PM**
Page 22: [5] Deleted	**Adams, Liz**	**6/3/24 5:34:00 PM**
Page 22: [5] Deleted	**Adams, Liz**	**6/3/24 5:34:00 PM**
Page 22: [5] Deleted	**Adams, Liz**	**6/3/24 5:34:00 PM**
Page 22: [5] Deleted	**Adams, Liz**	**6/3/24 5:34:00 PM**

Page 22: [5] Deleted	**Adams, Liz**	**6/3/24 5:34:00 PM**
Page 22: [6] Deleted	**Adams, Liz**	**5/26/24 12:49:00 PM**
Page 22: [7] Deleted	**Adams, Liz**	**5/26/24 12:59:00 PM**
Page 23: [8] Deleted	**Adams, Liz**	**5/26/24 1:00:00 PM**
Page 23: [8] Deleted	**Adams, Liz**	**5/26/24 1:00:00 PM**
Page 23: [8] Deleted	**Adams, Liz**	**5/26/24 1:00:00 PM**
Page 23: [9] Deleted	**Adams, Liz**	**5/26/24 11:57:00 AM**
Page 23: [10] Deleted	**Adams, Liz**	**6/3/24 5:41:00 PM**
Page 23: [10] Deleted	**Adams, Liz**	**6/3/24 5:41:00 PM**
Page 24: [11] Deleted	**Adams, Liz**	**6/14/24 4:52:00 PM**
Page 28: [12] Deleted	**Adams, Liz**	**6/17/24 7:40:00 PM**
Page 28: [13] Deleted	**Adams, Liz**	**6/17/24 7:49:00 PM**
Page 28: [14] Deleted	**Adams, Liz**	**6/17/24 7:39:00 PM**
Page 28: [15] Deleted	**Adams, Liz**	**6/17/24 7:31:00 PM**
Page 28: [16] Formatted	**Adams, Liz**	**6/17/24 4:51:00 PM**

Font: (Default) Times New Roman, 10 pt, Font color: Black, Pattern: Clear