Viability of Cloud Computing for Real-Time Numerical Weather Prediction

DAVID SIUTA, GREGORY WEST, HENRYK MODZELEWSKI, ROLAND SCHIGAS, AND ROLAND STULL

The University of British Columbia, Vancouver, British Columbia, Canada

(Manuscript received 18 April 2016, in final form 20 September 2016)

ABSTRACT

As cloud-service providers like Google, Amazon, and Microsoft decrease costs and increase performance, numerical weather prediction (NWP) in the cloud will become a reality not only for research use but for real-time use as well. The performance of the Weather Research and Forecasting (WRF) Model on the Google Cloud Platform is tested and configurations and optimizations of virtual machines that meet two main requirements of real-time NWP are found: 1) fast forecast completion (timeliness) and 2) economic cost effectiveness when compared with traditional on-premise high-performance computing hardware. Optimum performance was found by using the Intel compiler collection with no more than eight virtual CPUs per virtual machine. Using these configurations, real-time NWP on the Google Cloud Platform is found to be economically competitive when compared with the purchase of local high-performance computing hardware for NWP needs. Cloud-computing services are becoming viable alternatives to on-premise compute clusters for some applications.

1. Introduction

a. Background and motivation

Many government agencies, university groups, and private-sector companies run computationally expensive numerical weather prediction (NWP) models, some in real time. Some universities run daily NWP models under contract to bring in funding that pays for student salaries and hardware, to produce forecasts for field campaigns and experiments, and for teaching. Two examples of universities that run real-time forecasts in the Pacific Northwest are the University of Washington in Seattle, Washington (Mass and Kuo 1998; Grimit and Mass 2002), and The University of British Columbia (UBC) in Vancouver, British Columbia, Canada (Thomas et al. 2002; McCollor and Stull 2008). These institutions have made daily real-time forecasts since 1995 and 1996, respectively.

UBC runs a daily multimodel ensemble suite from 13 NWP runs initialized at 0000 UTC, with each run containing one or more two-way nested domains. We use output from each nested domain as an ensemble member. In total, these 13 NWP runs form a 39-member ensemble forecast at 0000 UTC when using the output from the individual domains. Eight of the 13 runs are made on our on-premise high-performance-computing (HPC) cluster. This HPC cluster, however, is aging, and we have begun to debate replacement strategies.

For smaller entities, such as our research team at UBC, the initial barrier to entry into the NWP forecast enterprise historically has been the high cost of computer hardware. Upfront costs include the purchase of on-premise HPC clusters and redundant systems for reliability, dedicated IT personnel salaries, and more recently the need to make ensemble forecasts (which require more computer power). In addition, HPC clusters typically have life spans on the order of three to five years, so any cluster eventually has to be replaced. Replacing old clusters with new clusters requires the same large upfront cost and personnel time. With the rise of private cloud-computing services such as those provided by Amazon, Google, Microsoft, and others, we decided to evaluate whether a cloud-computing-based solution could replace our HPC cluster and would work for our real-time NWP needs.

One of the advantages of cloud computing is its pay-per-use cost structure. This allows smaller payments over time instead of making a large equipment purchase

Denotes Open Access content.

Corresponding author e-mail: David Siuta, dsiuta@eos.ubc.ca

DOI: 10.1175/WAF-D-16-0075.1

up front. This is particularly well suited for university researchers who initiate real-time daily forecasts or individual research simulations that can be paid from typical modest research grants, instead of requiring a special large equipment grant up front. In addition, cloud services provide increased hardware reliability and easy expandability due to vast available resources. If more resources are desired, or if a physical machine goes down at a cloud service, more are always available for use. These two aspects in particular are appealing for real-time NWP use.

However, not all "clouds" are equal. The primary differences between cloud services are the cost structure, underlying network infrastructure, and computing resource types available. Because of this, some clouds may be better suited for real-time NWP than others. For example, Amazon and Microsoft charge for computing resources by the hour while Google charges by the minute.

Molthan et al. (2015) showed that NWP in the cloud is possible, but their costs were not economically advantageous compared to many small in-house systems such as those run by UBC. The Molthan et al. (2015) results reported a cost of $40–$75 for a 48-h deterministic simulation over the Gulf of Mexico. Extrapolating these costs to a typical 5-day forecast horizon (120 h) yields a cost of $98–$186 per forecast simulation. Making a once-daily forecast run out to 5 days would cost $36,000–$68,000 per year. While the Molthan et al. (2015) domains are different than the forecast domains UBC produces in real time, their size is similar. For comparison, the approximate cost per year of an on-premise HPC cluster is $29,000–$75,000 (plus any expenses for facility usage or IT personnel salary) per year. This latter estimate is based on the costs for a new, small (336–448 core), on-premise HPC cluster similar to that used at UBC (detailed later) costing on the order of $143,000–$226,000 (Lenovo 2015, personal communication), amortized over its typical life span (from three to five years). This cost quote, however, is for an on-premise HPC cluster capable of running UBC's entire 13-model (39 member) ensemble suite. Running the same 13 models (39 members) in the cloud using the extrapolated costs from the Molthan et al. (2015) study (which were for a single model simulation) would be significantly more expensive than local-HPC hardware (multiplying a cost of $98 per model simulation by 13 model simulations per day and 365 days per year yields a cost over $400,000).

The cloud-computing landscape, however, is changing rapidly. Since the publication of Molthan et al. (2015), increased competition between various cloud-computing services has lowered usage fees, changed cost structures, and expanded the type of services provided. The use of cloud-computing services within atmospheric sciences has likewise increased in the past few years. Vance (2016) provides an up-to-date summary of cloud-based research. In addition to NWP, cloud services are now being used for a variety of purposes, including data dissemination, online visualization tools, and web serving.

For example, Unidata is experimenting with moving several products to cloud systems, including the hosting of real-time data on their Thematic Real-Time Environmental Distributed Data Services (THREDDS; Ramamurthy 2016). Additionally, Unidata has developed cloud-based solutions for their Integrated Data Viewer (IDV) visualization product, and a slimmed-down version of the Advanced Weather Interactive Processing System version 2 (AWIPS2) that accesses resources stored on cloud servers. Ramamurthy (2016) mentions that part of the motivation behind Unidata's cloud migration is to satisfy increased data availability needs and also to promote faster research and ease of collaboration.

Similarly, the National Oceanic and Atmospheric Administration's (NOAA) Big Data Project (https://data-alliance.noaa.gov) is improving access to NOAA's large datasets from cloud platforms. Furthermore, if you are in a location that has slow or unreliable Internet access (like many developing nations), but still need to transfer large datasets (e.g., initial and lateral boundary conditions used for WRF), the cloud can alleviate data transfer issues since cloud providers can likely provide a faster and more reliable connection to the NWP initial-condition sources (McKenna 2016). This will result in overall faster forecast production. McKenna (2016) discusses how the costs of running a real-time forecast system in Amazon's cloud have fallen as speeds have increased.

The abundance of cloud-based research in the recent literature is clear evidence of a trend toward using cloud resources within the atmospheric sciences. Still, using cloud resources for real-time NWP is new and Molthan et al. (2015) and McKenna (2016) are the only studies that approach this topic. Our study highlights several best practices for running real-time NWP on cloud resources that have not yet been explored. In the next subsection, we summarize basic cloud-computing terminology and present the goal of our experiments.

b. *Cloud-computing terminology*

Understanding cloud-computing terminology and cost structures can be challenging. Here, we introduce terminology relevant for running WRF on the Google Cloud Platform (GCP) Compute Engine and compare terminology for traditional on-premise HPC systems. We chose to use the GCP because of the minute-based cost structure and the underlying network infrastructure

(network specifics are proprietary, but some details are available online at https://cloudplatform.googleblog.com/2015/06/A-Look-Inside-Googles-Data-Center-Networks.html). Some cloud providers may use slightly different terminology, but general descriptions of cloud components can be found in online documentation (Google 2016).

Cloud computing itself can be defined as when a user accesses a service, product, or application that is run within a virtual environment (Voorsluys et al. 2011). In our case, this virtual environment is provided by Google. There are four primary types of cloud services available: Software as a Service (SaaS), Platform as a Service (PaaS), Infrastructure as a Service (IaaS), and Everything as a Service (XaaS; Chang et al. 2010; Yuan 2016). The GCP Compute Engine that we use is an example of an IaaS, where Google provides raw computing resources at a per-use cost rate.

A virtual environment provided by an IaaS is similar to a computing system that one might design using local hardware. In the cloud, the user selects an operating system, specifies hard disk space and memory amounts, and specifies the number of compute cores [known as virtual central processing units (vCPUs)]. The virtual environment is run on hardware at the cloud provider's facilities and is accessed remotely by the user through a web browser or terminal. Collectively, once a virtual environment has been designed and turned on for use, it becomes known as a virtual machine (VM).

VM size (number of vCPUs) and memory amount are customizeable. We varied the amount of vCPUs per VM as part of our tests, as described in the next section. At the time of this writing, the GCP allowed between 1 and 32 vCPUs per VM, with larger machines being more expensive. Also, machines can have between 0.9 and 6.5 GB of memory per vCPU core, with more memory being more expensive. Accordingly, the largest 32-vCPU VM can have between 28.8 and 208 GB of memory—more than adequate for our WRF simulations.

The range of combinations of vCPUs and memory amounts creates an array of possible usage costs, adding to the complexity of using cloud resources cost effectively. Users must be careful to pick the configuration that best fits the application. For example, using a high-memory configuration will cost approximately twice as much as a low-memory configuration for VMs containing equivalent amounts of vCPUs. Likewise, for two VMs containing equivalent amounts of memory per vCPU, doubling the amount of vCPUs doubles the price (on the GCP). Charges for VM use are only accrued while a VM is turned on (on a per-minute basis). However, any disk storage is billed on the amount of GB used per month. Our specific VM configurations and the cost structure in place at the time of testing are detailed in section 2 (methods).

WRF requires a Linux or UNIX operating system, for which we chose CentOS because it is the same operating system we use for our on-premise HPC cluster. We could have picked another Linux operating system like Ubuntu or Red Hat, both of which are available as options on the GCP.

Once a VM has been designed, the user is free to install any libraries needed through either the operating system's standard package manager, or by manual installation. WRF requires the installation of netCDF, PNG, Jasper, zlib, and one of many available C and FORTRAN compilers [GNU, Portland Group International (PGI), Intel, etc.]. In our case, compilers and dependent libraries do not come preinstalled, nor are they provided by Google. Our use of the GNU compilers required manual installation, while use of the Intel compilers required us to obtain our own Intel license. To run WRF in parallel (which allows us to achieve faster simulation times by distributing a WRF run over multiple computing resources), Message Passing Interface (MPI) libraries are also required.

One of the benefits of using a cloud service is that once a VM is designed and all desired libraries installed, snapshots of the VM can be saved and deployed on additional VMs without the need to reinstall libraries and operating systems. This feature makes it easy to set up a virtual HPC cluster. A virtual HPC cluster is a group of VMs that communicate with each other in a fashion similar to compute nodes (motherboards or blades in old terminology) in a traditional on-premise HPC cluster, where each compute node has multiple cores, its own memory, disk storage, etc.

A single virtual HPC cluster can be used for a single WRF run, or additional runs can be handled on the same cluster by simply adding more VMs to the cluster. Virtual HPC clusters can also be replicated, if needed, through the use of snapshots. This would allow for multiple real-time WRF systems to be run in parallel (e.g., large-ensemble prediction systems). As part of our tests, we evaluate how the number of VMs used in a virtual cluster affects the model run time.

As charges for any VMs are accrued only while the VM is turned on, idle resources should be turned off when not in use. The GCP provides command-line tools that can be used to automate the turning on and off of both VMs and entire virtual HPC clusters to decrease costs.

The primary purpose of this study is to test the viability of using the GCP for our real-time NWP needs at UBC. We evaluate the GCP under their most recent pricing plan (as of September 2016) to see if our optimizations make the GCP an economically viable

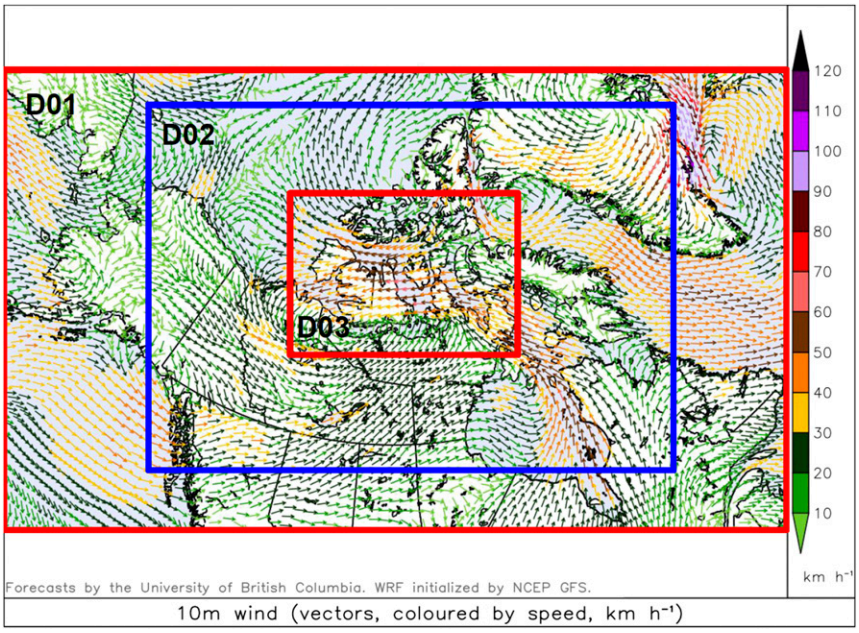

FIG. 1. WRF-Arctic domains. Grid lengths are 36, 12, and 4 km for D01, D02, and D03, respectively. Horizontal grid counts are 200 × 108 (D01), 346 × 250 (D02), and 448 × 307 (D03). All domains have 41 layers in the vertical. Sample 10-m wind output is shown as the colored vectors valid at 1800 UTC 26 Sep 2015.

replacement for our aging on-premise HPC cluster. We build upon Molthan et al. (2015) and McKenna (2016) as follows: while Molthan et al. (2015) vary the number of VMs, we also optimize the number of vCPUs per VM, the amount of memory per vCPU, and compiler choice; additionally, we consider preemptible runs (which offer lower cost, but with the threat of being canceled). Also, while McKenna (2016) report success in repeating an operational forecast system in the cloud, our study is the first to compare costs for real-time NWP in the cloud to that of a local hardware system.

While the optimal configuration and the resulting cost depends on the model domain size, it is thought that our regional WRF domains detailed here are typical for many entities running WRF in real time. Thus, our results may apply to many WRF users, or at the very least motivate them to investigate for themselves using a similar methodology. Section 2 describes the research methodology. Section 3 provides a discussion of the results, followed by conclusions and suggestions for future work in section 4.

2. Methods

a. *WRF model case-study configuration*

A WRF (version 3.7.1; Skamarock et al. 2008) model run using the Advanced Research WRF (ARW) core was performed as a benchmark test covering a large area of the Canadian Arctic with nested grids (Δx = 36, 12, and 4 km; 41 vertical levels; Fig. 1). The domain setup in Fig. 1 was chosen for our experiment because it is currently one of the most computationally expensive real-time runs employed at UBC on our local hardware, because of the large area covered by the 4- and 12-km grids. This makes it a good test for our evaluation of the GCP and, hopefully, for many other entities that run similarly sized regional domains. This simulation will be referred to throughout as WRF-Arctic. The forecast horizons for the 36-, 12-, and 4-km grids were 156, 108, and 60 h, respectively. WRF-Arctic used the Yonsei University planetary boundary layer scheme (Hong et al. 2006), the MM5 similarity surface layer scheme (Zhang and Anthes 1982), the Noah land surface model (Ek 2003), the WRF single-moment 5-class microphysics scheme (Hong et al. 2004), the Rapid Radiative Transfer Model longwave radiation scheme (Mlawer et al. 1997), and the Dudhia shortwave radiation scheme (Dudhia 1989).

The adaptive time step (Hutchinson 2007) in WRF was enabled, where the time step is allowed to vary to optimize NWP computation speed while continually satisfying the Courant–Friedrichs–Lewy condition (Warner 2011; Skamarock et al. 2008). Use of the adaptive time step allows runs to finish faster when the weather is calmer, such as when an anticyclone dominates in the WRF domain. For the domains shown in Fig. 1, use of the

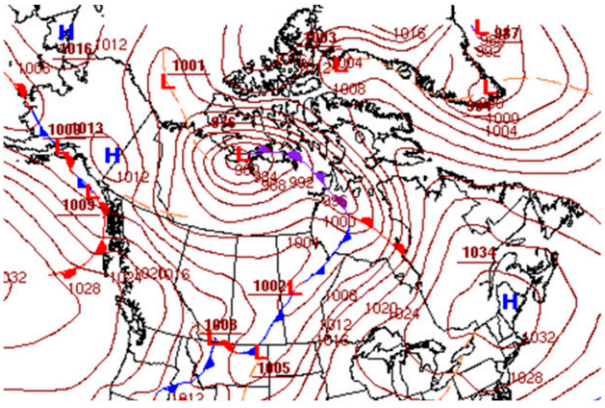

FIG. 2. The 1800 UTC 26 Sep 2015 surface analysis from the NOAA/NWS Weather Prediction Center showing a 97.6-kPa low over the Northwest Territories, transiting our WRF-Arctic domains.

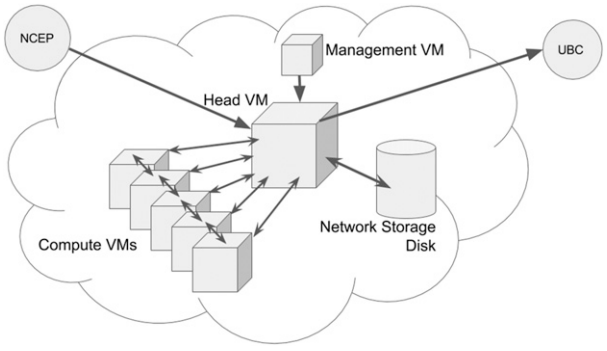

FIG. 3. Sample diagram of a virtual HPC cluster on the GCP used in this study. Blocks represent individual VMs. Cylinders represent attached network storage disks. UBC represents local hardware at the University of British Columbia. Arrows represent communications pathways. Table 1 details the base costs for each of these components and the equivalent terminology used by Google.

adaptive time step on UBC's on-premise HPC cluster allows a 156-h forecast to finish in about 100 min during quiescent weather compared with about 150 min during active weather. When similar computations are done on cloud-based computers where billing is by the minute, savings due to adaptive time stepping can be substantial. For the 4-km domain in this case study, we found the time steps to range from 8.3 to 66 s, but they were mostly around 30 s. While the adaptive time-step option works well for our use, there may be situations where consistent completion times are required, making its use nonviable. Nonetheless, it has the potential to reduce costs to some cloud users by capitalizing on shorter run times during calm weather regimes.

We tested WRF-Arctic runs on the GCP for a case study initialized from the 0.5° Global Forecast System (GFS) at 1200 UTC 25 September 2015. This case study includes the rapid development of a surface low in northern Alberta/Northwest Territories on 25–26 September 2015. The low slowly transits the model domains (Fig. 1) through the forecast period. The NOAA/NWS Weather Prediction Center's 1800 UTC 26 September 2015 analysis (Fig. 2) shows this low at its maximum strength (97.6 kPa) situated over the Northwest Territories, when the low is in the middle of our forecast domains. In a manner similar to Molthan et al. (2015), we will extrapolate the costs of this one case-study run to estimate the annual costs of this single simulation.

We also compare simulation times for the case study to those obtained on UBC's on-premise HPC cluster. UBC's on-premise HPC cluster is a 448-core, multinode cluster using Intel Xeon E5 processors with Sandy Bridge architecture and IBM hardware and Infiniband interconnect (we use 176 cores to produce this simulation). UBC's cluster also uses the commercially available PGI C and FORTRAN compiler collection.

b. *Virtual HPC cluster configuration*

We implemented a hybrid approach on the GCP. That is, the most computationally expensive parts of the forecast system (the preprocessing and model simulation) were done on the GCP. The raw model output was transferred to UBC where local hardware and postprocessing software already in place were used for model output archival, graphics production, database storage, product verification, and web serving. While our postprocessing system is not as computationally expensive as the model simulation, the amount of different products generated make transferring this process to the GCP beyond the scope of this investigation. We plan to reproduce our postprocessing system in the cloud for future investigations.

Figure 3 shows a schematic of our virtual HPC cluster design on the GCP. We follow a method similar to that of Molthan et al. (2015), where we configure a single VM as a head node (Head VM) that controls the cluster communications between other VMs used as compute nodes [Compute VMs (CVM)]. The Head VM is also attached to a 500-GB network storage disk via network file system (NFS) mounting. All preprocessed input data and WRF output are written to this disk. The Head VM is responsible for retrieving the GFS initial and lateral boundary condition grids from the National Centers for Environmental Prediction (NCEP) ftp server and running the WRF Preprocessing System (WPS). The Head and Compute VMs are turned off (in a virtual sense) when each forecast is finished, to save costs. Table 1 details the machine types used for each component in Fig. 3 and also the base costs as provided by Google (2016).

TABLE 1. Cost summary (September 2016) of the virtual HPC cluster components in Fig. 3. The numbers of CVMs and vCPUs varied for our case-study experiments; *N* is the number of vCPUs per VM.

Component	Machine used	Base cost [U.S. dollars (USD)]
Management VM	f1-micro	0.0056 per hour
Head VM	n1-highcpu-8	0.304 per hour
Compute VM	n1-highcpu-*N* vCPUs	0.038 per vCPU per hour
Network storage disk	Standard persistent disk	0.04 per GB per month

For our investigation, the Head VM required 10 GB of local disk space to install the CentOS Linux operating system and required libraries for WRF. This setup is saved as a snapshot, (i.e., a saved VM configuration) that can be replicated onto any new VMs needed. In our case, each CVM is a direct replica of the Head VM in terms of the operating system, local storage space, and installed libraries. A small, less expensive Management VM (Fig. 3) is always running and is used to turn on and off the Head VM between forecast simulations.

Once the initial and lateral boundary conditions are downloaded, the WPS and real.exe processes are run on the Head VM to create a balanced initial state for WRF. The combined initial-condition download, WPS, and real.exe processes take about 1 h to finish and do not vary much from run to run for the same domains. We always allot 1 h for these processes in our economic calculations. Once the WPS process has been completed, the Head VM turns on any specified number of CVMs to run the WRF simulation. Each CVM and the Head VM must be configured to communicate with each other by Secure Shell (ssh), making sure that each CVM has the same known_hosts file containing the unique names and local network addresses of all CVMs used as well as for the Head VM. If this is not done, the MPI communications used by WRF for parallel runs will not work. While the actual physical machine on which each CVM runs will change each time it is turned on, the local network address remains static, allowing this setup to work.

Several different versions of MPI are available. We employ MPICH MPI when using WRF compiled with the GNU compiler collection, and Intel MPI for WRF compiled with the Intel compiler collection.

Users of the GCP have some control over the physical hardware on which a virtual environment is run. On the GCP, computing resources are split into regions and zones. Regions are computing facilities located in specific areas of a country/continent. Zones are different computing resources within a single region. Zones are grouped by processor type. We use VMs that are always located within the same zone of the GCP, so that the MPI process will work (central U.S. region, zone C, Intel Haswell processors). When using these processors, a vCPU is defined as a single hardware hyper-thread (details available online at https://cloud.google.com/compute/docs/machine-types).

The Head VM is scripted to turn on any number of CVMs needed for the simulation. While the number of vCPUs per CVM is varied as part of our tests, we always use machines that are of type highcpu according to Google (2016). These types of VMs have less memory available than standard or highmemory VM types. Because highcpu machines offer less memory, their use results in large cost savings, especially over continued use. We found the amount of memory available on highcpu machines to be suitable for WRF (0.9 GB of memory per vCPU).

c. *Experiments*

Several computing configurations were tested to determine GCP's efficacy as a real-time forecast virtual HPC cluster, with the goal of reducing simulation time and cost. First, scaling was tested by simulating the WRF-Arctic case study using eight-vCPU VMs as CVMs. Scaling is the ability of the simulation to finish faster through the use of additional computing power (more CVMs or total vCPUs). Scaling is limited by the communication speed between CVMs and also between vCPUs on an individual CVM. On the GCP, communication speed between VMs is a function of the number of vCPUs on a VM (with peak speeds reached at eight vCPUs), and disk-write speeds are a function of disk size (larger disks provide faster write speeds).

For our experiment, additional groups of eight CVMs were added until simulation run time stopped decreasing. Second, we compared simulation run times using two different compilers (GNU and Intel). Finally, the number of vCPUs per CVM was varied while holding the total number of vCPUs as constant as possible. VM sizes can range from 1 to 32 vCPUs with several memory options, for which we always found the lowest memory option to be suitable. Table 2 summarizes these tests, including the factors that were changed and held constant. A small section of the tests were completed more than once to check for run-time consistency (not shown). We found simulation times to be consistently within 2–3 min of each other, and therefore the results presented here are from single tests and not an average of multiple tests.

One of the major costs with cloud computing is associated with NWP model testing, which would not be accrued by using already-owned local hardware. To this

TABLE 2. Summary of the tests performed. Simulation costs include preprocessing but not data egress.

Test group/ID	Compiler used		Compute VM setup and results				
	GNU	Intel	Total No. of VMs used	No. of vCPUs per VM	Total No. of vCPUs	Simulation time (min)	Simulation cost per year (USD)
Scale VMs with GNU compiler							
GNU1	X		8	8	64	320	5,729
GNU2	X		16	8	128	292	9,586
GNU3	X		24	8	192	250	11,967
GNU4	X		32	8	256	220	13,838
GNU5	X		40	8	320	207	16,111
GNU6	X		48	8	384	228	21,079
Scale VMs with Intel compiler							
Intel1		X	8	8	64	188	3,532
Intel2		X	16	8	128	152	5,185
Intel3		X	24	8	192	141	6,928
Intel4		X	32	8	256	139	8,895
Intel5		X	40	8	320	149	11,713
Adjust vCPUs per VM							
CPU1		X	32	4	128	157	5,336
CPU2		X	16	8	128	152	5,185
CPU3		X	11	12	132	158	5,511
CPU4		X	8	16	128	170	5,741
CPU5		X	5	24	120	206	6,491
CPU6		X	4	32	128	450	14,549

extent, testing of model configurations (such as the performance of particular physical parameterization schemes, grid lengths, and initial-condition sources) may be more beneficial on local hardware, if it exists, since there would be no extra associated costs. After testing, the best grid length, physical parameterization, and initial-condition choices can be migrated to the cloud system for real-time implementation. Since our main concern was achieving timely forecasts with feasible economic cost for our needs, an evaluation of our forecast performance was beyond the scope of this paper. We instead use our currently operational WRF-Arctic configuration for our case study.

Google allows for 80% cost savings by using preemptible VMs. The only difference (other than cost) between a preemptible VM and a nonpreemptible VM is that preemptible VMs can be shut down by Google if the resource is needed by other users. This means that any process running on a preempted VM is killed. When not preempted, the simulation times of preemptible and nonpreemptible machines were the same. Currently, preemptible VMs do not have any delay in start time (i.e., they turn on immediately when requested), but can be preempted at any time after being turned on. As preemptible runs cannot be tolerated for real-time NWP (when hard deadlines exist), we will not consider their price when calculating the cost of a real-time NWP suite in the cloud. We mention the preemptible machines because they could be of use to academic researchers acting on smaller budgets. In addition, these VMs could be used for the NWP model testing mentioned above if local systems do not exist, or if local hardware is not powerful enough for the task. Ways of using preemptible machines for real-time NWP could be explored in the future, for example, through redundant systems, or large ensembles that might be less affected if one or a few ensemble members fail to complete.

After our tests were completed, costs were estimated using the pricing as of September 2016. Current pricing is provided online (https://cloud.google.com/pricing/). For an entirely preemptible system, the costs quoted in Table 2 would be reduced by 80%.

3. Results and discussion

The first set of tests (GNU1–GNU6; Table 2), aimed at assessing scalability, were completed using the free GNU compiler collection. Starting with eight CVMs (8 vCPUs each, 64 total vCPUs), groups of eight CVMs were added in each test run until the simulation time stopped decreasing (Fig. 4). The fastest simulation time achieved for this test was 207 min using 40 CVMs (GNU5; Table 2).

At 40 CVMs (GNU5), a total of 320 vCPUs are individually communicating with each other through the MPI process. This is 5 times more network communications than when using the original eight CVMs (GNU1). It is possible that this amount of communication pathways has likely reached the cap of Google's current networking infrastructure. Additional network

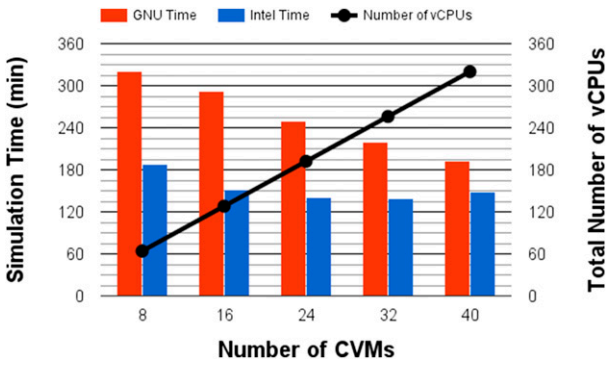

FIG. 4. Scaling of the WRF-Arctic compiled with GNU and Intel compiler collections. The red and blue bars represent the total simulation times for various numbers of CVMs for the GNU and Intel compilers, respectively. The black line indicates the total number of vCPUs.

traffic past 40 CVMs results in slower simulation times. Specific details on the GCP proprietary networking infrastructure are not publically available, but the GCP relies on an underlying Ethernet network without Infiniband interconnect.

Tests GNU1–GNU5 show that the GCP has some ability to scale for parallelized HPC applications. Users can add more CVMs as needed if faster simulation times are required, but there is a limit to decreasing WRF simulation times and what is cost effective. However, the 207-min simulation time achieved with 40 CVMs is longer than when run on UBC's on-premise HPC cluster (121 min). Three important differences should be noted between UBC's HPC cluster and the configuration of the GCP VMs used in the first batch of tests: 1) GCP VMs run on newer Intel Haswell Architecture processors; 2) GCP VMs use the slower, but free, GNU compiler collection; and 3) GCP VMs use Ethernet communications. Nonetheless, we find that HPC-like performance on the GCP is possible. The cost of running 40 CVMs for 207 min, however, makes this setup relatively expensive for daily real-time use (costs extrapolate to over $15,000 per year for a daily WRF-Arctic simulation).

The Intel compiler collection has been shown to result in up to a 42% improvement in speed over the GNU compiler (HPC Advisory Council 2012; Langkamp and Böhner 2011). We reran the scaling tests by recompiling WRF using the 2016 Intel compiler collection (Intel1–Intel5; Table 2). To use the Intel compilers, we had to obtain a license from Intel and install the software package locally on our VMs prior to recompiling WRF. Since the physical hardware on the GCP uses Intel processors, the Intel compiler collection can better optimize WRF during compilation (when compared to the GNU compilers) to take advantage of the Intel processor architecture. This is the primary difference between using WRF compiled using GNU compilers and Intel compilers and can lead to faster execution. While other commercial compilers are available, our choice of using the Intel collection was due to its support by WRF and the availability of a free evaluation license from Intel.

The difference in performance between the Intel-compiled WRF and the GNU-compiled WRF is significant. The Intel-compiled version of WRF consistently provided a 45%–50% speed-up over the GNU-compiled WRF for identical scaling tests (Fig. 4). The fastest simulation time when using the Intel compilers was 139 min when using 256 vCPUs (Intel4; Table 2). By achieving a 139-min run time using the commercially available Intel compiler collection, the gap between the simulation time achieved on UBC's local cluster and that on the GCP is narrowed to only 18 min (down from an 86-min gap when using the GNU compilers). Thus, we conclude that the use of the GNU compilers is the primary reason for the large difference between the best simulation time achieved using the GNU compilers (GNU5) and the simulation time achieved on UBC's local cluster (i.e., the lack of Infiniband interconnect on the GCP is a much smaller factor).

Also, fewer CVMs are needed when using the Intel compilers to exceed the best performance found with the slower GNU compilers (8 versus 40 total CVMs, respectively). Because Google charges are based on both the amount of virtual hardware and the usage time, the Intel compilers result in significant cost savings by allowing simulations to complete faster and on fewer CVMs. While the cost of an Intel compiler license can vary by user ($1,000 per year for the renewal of a single-user license that costs $3,000 initially) and the purchase is the responsibility of the user (UBC in this case), for most use cases the costs will quickly be recovered by the resultant savings (shown later).

Next, tests CPU1–CPU6 (Table 2) were performed to determine the influence of the number of vCPUs per CVM on simulation time. The amount of vCPUs per CVM was varied while keeping the total vCPUs used for the simulation as close to 128 as possible. For tests with greater than eight vCPUs per CVM, simulation times slowed (Fig. 5). The slowdown is greatest when using the largest 32-vCPU VMs. There is limited network throughput that can become clogged with additional traffic as a result of using more vCPUs per CVM (each vCPU is required to communicate with vCPUs on CVMs containing neighboring grid points). This result shows that using the most powerful VMs available, in terms of the amount of vCPUs per CVM, is not necessarily better than using a

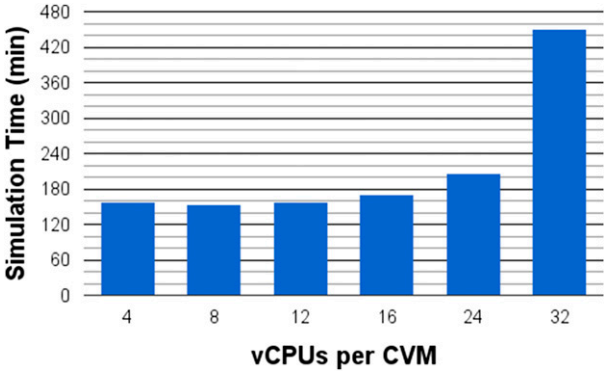

FIG. 5. Evaluation of the impact of changing the number of vCPUs per CVM on simulation time while using the Intel-compiled WRF. The total number of vCPUs summed over all virtual machines used during the simulation was kept as close to 128 as possible.

larger number of smaller CVMs. Using the larger CVMs can lead to oversubscription of network resources on the VM, slowing simulation times. The eight-vCPU VMs were the best CVM choice for NWP-run efficiency, and thus cost, for our case study on the GCP.

Scaling as we have tested it is also dependent on the design of the WRF code and domain size. Different domains will exhibit different scaling characteristics, and some discussion of the reason behind this is provided online (https://www2.cisl.ucar.edu/software/community-models/optimizing-wrf-performance-yellowstone). While our domains are unique to UBC, our domain setup is typical of other real-time NWP forecasts covering regional areas (e.g., those created by the University of Washington), and thus our results should generalize to users of similar-sized domains. We also find that scaling is much poorer (simulation time stops decreasing much sooner through the addition of CVMs to the virtual HPC cluster) when using the Intel compilers for our domains, even though the overall simulation times were much faster.

Case-study simulation times for our domains were extrapolated to estimate the cost of daily runs for a year, by multiplying the cost for a single run by 365 days (Table 2 and Fig. 6). The estimate takes into account differences in compiler choice and the number of CVMs, using the optimum eight vCPU per CVM setup. Costs include an hour for preprocessing of the WRF initialization grids, and the small, but always running, Management VM that automatically turns on and off the Head VM. In addition, costs include a 500-GB network storage disk attached to the Head VM and 10-GB system disks on each of the CVMs. We provide two cost estimates for each setup: 1) without data egress and 2) with 45 GB of daily data egress for the hourly model output to UBC.

While this study focused on simulation time, it is important to note the cost of data egress (transfer of data to external systems, i.e., from the GCP to UBC). Costs can be significant for users (such as UBC) transferring the

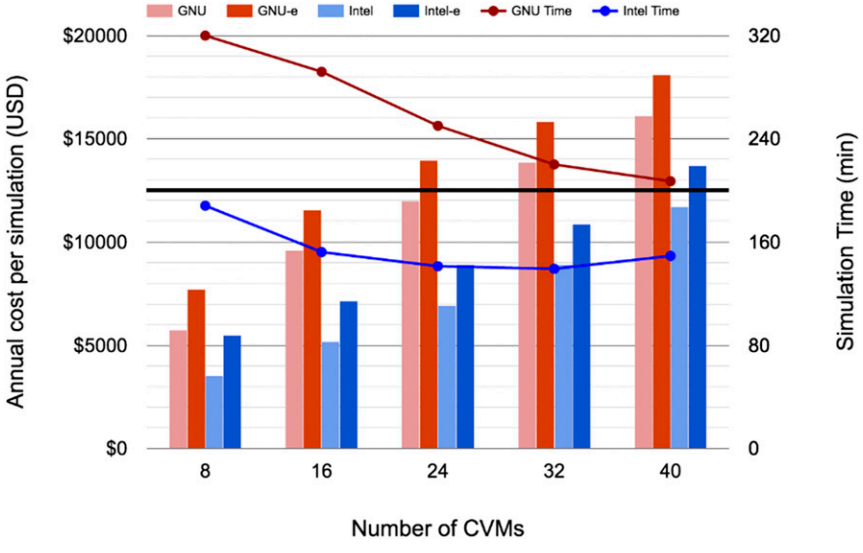

FIG. 6. Annual cost comparison for daily WRF-Arctic simulations using different compiler collections and number of CVMs with eight vCPUs per CVM. Annual costs are estimated by multiplying the cost of our 1-day (with 156-h forecast horizon) case study by 365 days. Bars correspond to the cost axis at left; lines correspond to the simulation time axis at right. The -e suffix in the legend denotes a cost estimate that includes the egress of hourly raw model output. The boldface black horizontal line represents a 200-min simulation time.

raw model output files from the cloud to local systems at \$0.04–\$0.12 per GB. Data egress costs are controlled by the amount of data transferred (more transfer is discounted) and any networking peering agreements in place with Google (direct connection to Google's network). Since the output for the WRF-Arctic simulation totals 45 GB per run, egress costs are between \$1.80 and \$5.40 per simulation, or \$657–\$1971 per year (denoted by -e suffix; Fig. 6). Network ingress (transfer of data from external systems to the GCP, i.e., downloading initial conditions from NCEP) has no charge (Google 2016). Because of egress costs, users might decide to do all postprocessing and web serving in the cloud, instead of on local, user-owned hardware. We used the maximum possible cost in our estimates that include data egress.

As an example of cost savings due to compiler choice, we compared the costs required to run the case-study simulation in approximately 200 min (Fig. 6, thick black line). It would take 40 CVMs (GNU5; Table 2) to come close to this time with the GNU compiler, but only eight CVMs (Intel1; Table 2) with the Intel compiler. Including data egress, annual costs for achieving simulations of approximately 200 min for our domains are estimated at \$5,503 using the Intel compilers (Fig. 6, dark blue bar on the left side), whereas GNU costs are \$18,082 (Fig. 6, dark red bar on the right side). Because fewer CVMs are required with the Intel compiler, costs are reduced by approximately 70%. When not including data egress, the cost savings increase to roughly 80% (\$3,532 for the Intel-compiled version of WRF compared to \$16,111 for the GNU compiled version).

In section 1, the cost per year of a small, on-premise HPC cluster capable of running our entire ensemble suite was estimated to be between \$29,000 and \$75,000, plus any expenses for facility use or IT personnel salary. For the same costs, we could run approximately 6–15 daily equivalent WRF-Arctic simulations when including data egress (calculated using results of Intel1; Table 2), or 8–21 simulations without including data egress. While Intel1 did not provide the fastest simulation times, we use it for comparison because the ability to run all these simulations simultaneously on cloud resources ultimately would save more time than using a few additional CVMs for faster individual runs. Although personnel are required to provide system administration for both on-premise and cloud systems, additional costs for on-premise clusters include salary to manage hardware (which is common for small university groups and private companies), and any facility-use charges. Because of this, the estimates provided above are conservative; namely, more simulations could be afforded in some situations on a cloud platform relative to using local hardware.

While our results show that cloud-based NWP on the GCP is economically viable for our needs when compared to on-premise HPC clusters, cloud-based systems also have the added benefit of reduced impact of computer hardware failure, less time addressing hardware issues, and ease of making additional real-time forecast runs because of cloud expandability. Although we did not estimate costs for these intangible benefits, they could translate into substantial savings, for example, by not needing to purchase redundant hardware, avoiding penalties for missed NWP runs, or losing contracts from a lack of reliability. Additionally, evolving cloud networking infrastructure and specific offerings for HPC could alleviate some of the networking bottlenecks we encountered.

4. Conclusions

This study evaluated the performance of the GCP for real-time NWP using a WRF model simulation over the Canadian Arctic, a domain size that is typical of many regional modeling systems. For cloud-computing services to be a viable replacement for on-premise HPC hardware, forecast production must be timely, economically viable, reliable, and have enough resources to run ensemble prediction systems. We evaluated whether forecasts could be produced on the GCP to satisfy the first two requirements by searching for optimizations that allowed forecasts to be completed in a similar time to that obtained on our on-premise HPC hardware, and by comparing the annual cost of such runs to the purchase of a new, small on-premise HPC hardware cluster. The nature of cloud computing intrinsically satisfies the latter requirements.

The GCP completed a large benchmark simulation in a similar total run time as to an on-premise HPC cluster (with different technical specifications). The on-premise HPC cluster at UBC finished the simulation in 121 min while the GCP finished the simulation in 139 min at its fastest (Intel 4; Table 2), despite newer hardware. Nevertheless, the ability of the GCP to run large simulations in a similar time as on-premise HPC hardware makes it a serious contender for government agencies, private companies, and universities considering the cloud as a replacement for aging hardware.

Several factors have been shown to influence simulation time and overall cost. The most important factor was the compiler choice used during WRF installation, as evidenced through the cost comparison of tests GNU1–GNU6 with the equivalent tests Intel1–Intel5. When comparing the cost of achieving the fastest possible run time for this case study using the GNU compilers (GNU5; Table 2), the Intel compilers provided

about 70%–80% cost savings (Intel1; Table 2) primarily because 32 fewer CVMs were needed to achieve the same run time.

The number of vCPUs per CVM and the overall number of CVMs in the cluster also influenced simulation time and thus cost. The optimal CVM configuration found in this case study was to use the Intel-compiled WRF and the low-memory CVMs with eight vCPUs each (CPU2; Table 2). Using this configuration with 16 CVMs (Intel2; Table 2) provided the best compromise between simulation time and cost as using more CVMs increased cost while only marginally improving simulation time. With this optimized configuration, the total cost for once-daily simulations, including data egress, is approximately $7,000 per year. For users with less stringent run-time requirements, using only eight CVMs (Intel1; Table 2) resulted in simulation times only 30 min longer, while costing just over $5,500 for once-daily WRF-Arctic simulations per year (including data egress).

Our cost comparison shows that using the GCP as a replacement for our on-premise hardware is economically viable. We estimated the costs for an on-premise HPC hardware cluster similar to that used at UBC to be $29,000–$75,000 per year ($143,000–$226,000 amortized over three to five years), plus any IT personnel salary and facility-usage charges. Thus, this study showed that cloud-computing costs are competitive with, and can be lower than, a traditional setup using on-premise HPC hardware for our NWP needs. In addition, since we do not include the costs of local IT staff or facility-usage charges, which can vary greatly, our cost estimate of the number of cloud-based NWP runs that could be afforded for the same price as an on-premise system is conservative.

It is difficult to put a price tag on the added benefits of increased reliability and ease of expansion in cloud resources. However, these aspects of cloud computing make cloud resources useful to organizations running ensemble prediction systems because they offer the ability to make simultaneous runs on a nearly unlimited supply of CVMs. This allows for larger, more diverse ensemble systems made using more initial-condition sources, different model physics or dynamical cores, and grid lengths. In addition, cloud-computing services provide robust reliability for forecast centers because of the very large amount of available redundant hardware that is seamlessly swapped in as needed. Further, cloud computing could be beneficial for researchers who do not have access to an on-premise HPC hardware cluster or IT staff, but need to perform HPC-like calculations such as NWP.

While our results are unique to UBC's forecast domains, our conclusions about the use of Intel compilers and VM size can likely be translated to other clouds. At the minimum, users of other clouds should be aware that VMs can be configured to significantly reduce costs associated with running WRF if you first take the time to test VM sizes and compiler choices. We hope our explanation of cloud computing, and our results, will inspire NWP users to experiment and price out cloud computing for their needs.

Because we have shown that real-time NWP in the cloud is economical and provides timely output, we have started running one of our real-time NWP models daily on the GCP. As of the date this paper was written, we have run this forecast for approximately one year with 100% GCP reliability. This newly proven viability of cloud computing could indicate a turning point for real-time NWP, in that cloud computing may be a more affordable option for some.

While cloud computing has been shown here to have several benefits, there are still complications to overcome. These include costs associated with data egress and model output archival in the cloud (if desired), as well as the initial migration from the existing local-hardware system to the cloud. Cost structures set by cloud-computing providers are often complicated, and determining 1) the optimum, cost-minimized set up for each use case and 2) whether these optimized costs are viable will require testing by the individual user. In addition, local hardware purchased by grants will be available for use even after these grants expire at no extra cost, while cloud-computing costs will continue to be billed. One example where local hardware may be particularly beneficial is in the case of exploratory work by students, where mistakes are frequent and reruns on the cloud may cost more than reruns on local hardware. However, cloud-based solutions to this problem may exist in the form of highly discounted preemptible machines that offer an 80% discount.

Future work on using NWP in the cloud should investigate the effects of different domain sizes on the optimum setup. It is possible that much larger domains than the ones used here may have different optimum configurations. However, our results pinpoint several key aspects of cloud computing to be aware of when doing tests to find the optimum setup. Also, there are other pay-to-use compilers that were not tested here. Optimum compiler choice, vCPU count, and CVM scaling will likely vary by cloud provider as a result of each platform's unique hardware, communication infrastructure, and pricing.

Acknowledgments. The authors thank BC Hydro, Mitacs, and Natural Science and Engineering Research Council of Canada (NSERC) for providing the funds to

perform this research. Also, the help of several members of the Google Cloud Platform team including Jasen Baker, Marcos Novaes, David Engelbrecht, James Brown, Ben Menasha, Sami Iqram, Larry Quesada, and Miles Ward is greatly appreciated. We also thank the three anonymous reviewers for their comments and suggestions, which greatly improved this manuscript.

REFERENCES

Chang, W. Y., H. Abu-Amara, and J. F. Sanford, 2010: *Transforming Enterprise Cloud Services*. Springer, 428 pp.

Dudhia, J., 1989: Numerical study of convection observed during the Winter Monsoon Experiment using a mesoscale two-dimensional model. *J. Atmos. Sci.*, **46**, 3077–3107, doi:10.1175/1520-0469(1989)046<3077:NSOCOD>2.0.CO;2.

Ek, M. B., 2003: Implementation of Noah land surface model advances in the National Centers for Environmental Prediction operational mesoscale Eta Model. *J. Geophys. Res.*, **108**, 8851, doi:10.1029/2002JD003296.

Google, 2016: Google compute engine pricing. [Available online at https://cloud.google.com/compute/pricing.]

Grimit, E. P., and C. F. Mass, 2002: Initial results of a mesoscale short-range ensemble forecasting system over the Pacific Northwest. *Wea. Forecasting*, **17**, 192–205, doi:10.1175/1520-0434(2002)017<0192:IROAMS>2.0.CO;2.

Hong, S.-Y., J. Dudhia, and S.-H. Chen, 2004: A revised approach to ice microphysical processes for the bulk parameterization of clouds and precipitation. *Mon. Wea. Rev.*, **132**, 103–120, doi:10.1175/1520-0493(2004)132<0103:ARATIM>2.0.CO;2.

——, Y. Noh, and J. Dudhia, 2006: A new vertical diffusion package with an explicit treatment of entrainment processes. *Mon. Wea. Rev.*, **134**, 2318–2341, doi:10.1175/MWR3199.1.

HPC Advisory Council, 2012: Weather Research and Forecasting (WRF) performance benchmark and profiling. 17 pp. [Available online at http://www.hpcadvisorycouncil.com/pdf/WRF_Analysis_and_Profiling_7_2012.pdf.]

Hutchinson, T. A., 2007: An adaptive time-step for increased model efficiency. *Extended Abstracts*, *Eighth WRF Users' Workshop*, Boulder, CO, NCAR, 9.4. [Available online at http://www2.mmm.ucar.edu/wrf/users/workshops/WS2007/abstracts/9-4_Hutchinson.pdf.]

Langkamp, T., and J. Böhner, 2011: Influence of the compiler on multi-CPU performance of WRFv3. *Geosci. Model Dev.*, **4**, 611–623, doi:10.5194/gmd-4-611-2011.

Mass, C. F., and Y.-H. Kuo, 1998: Regional real-time numerical weather prediction: Current status and future potential. *Bull. Amer. Meteor. Soc.*, **79**, 253–263, doi:10.1175/1520-0477(1998)079<0253:RRTNWP>2.0.CO;2.

McCollor, D., and R. Stull, 2008: Hydrometeorological short-range ensemble forecasts in complex terrain. Part I: Meteorological evaluation. *Wea. Forecasting*, **23**, 533–556, doi:10.1175/2008WAF2007063.1.

McKenna, B., 2016: Dubai operational forecasting system in Amazon cloud. *Cloud Computing in Ocean and Atmospheric Sciences*, T. C. Vance et al., Eds., Academic Press, 325–345, doi:10.1016/B978-0-12-803192-6.00016-5.

Mlawer, E. J., S. J. Taubman, P. D. Brown, M. J. Iacono, and S. A. Clough, 1997: Radiative transfer for inhomogeneous atmospheres: RRTM, a validated correlated-k model for the longwave. *J. Geophys. Res.*, **102**, 16 663–16 682, doi:10.1029/97JD00237.

Molthan, A. L., J. L. Case, J. Venner, R. Schroeder, M. R. Checchi, B. T. Zavodsky, A. Limaye, and R. G. O'Brien, 2015: Clouds in the cloud: Weather forecasts and applications within cloud computing environments. *Bull. Amer. Meteor. Soc.*, **96**, 1369–1379, doi:10.1175/BAMS-D-14-00013.1.

Ramamurthy, M., 2016: Data-driven atmospheric sciences using cloud-based cyberinfrastructure: Plans, opportunities, and challenges for a real-time weather data facility. *Cloud Computing in Ocean and Atmospheric Sciences*, T. C. Vance et al., Eds., Academic Press, 43–58, doi:10.1016/B978-0-12-803192-6.00004-9.

Skamarock, W., and Coauthors, 2008: A description of the Advanced Research WRF version 3. NCAR Tech. Note NCAR/TN-475+STR, 113 pp., doi:10.5065/D68S4MVH.

Thomas, S. J., J. P. Hacker, M. Desgagné, and R. B. Stull, 2002: An ensemble analysis of forecast errors related to floating point performance. *Wea. Forecasting*, **17**, 898–906, doi:10.1175/1520-0434(2002)017<0898:AEAOFE>2.0.CO;2.

Vance, T. C., 2016: A primer on cloud computing. *Cloud Computing in Ocean and Atmospheric Sciences*, T. C. Vance et al., Eds., Academic Press, 1–13, doi:10.1016/B978-0-12-803192-6.00001-3.

Voorsluys, W., J. Broberg, and R. Buyya, 2011: *Cloud Computing: Principles and Paradigms*. John Wiley and Sons, 637 pp.

Warner, T. T., 2011: *Numerical Weather and Climate Prediction*. Cambridge University Press, 526 pp.

Yuan, M., 2016: Conclusion and the road ahead. *Cloud Computing in Ocean and Atmospheric Sciences*, T. C. Vance et al., Eds., Academic Press, 385–391, doi:10.1016/B978-0-12-803192-6.00020-7.

Zhang, D., and R. A. Anthes, 1982: A high-resolution model of the planetary boundary layer—Sensitivity tests and comparisons with SESAME-79 data. *J. Appl. Meteor.*, **21**, 1594–1609, doi:10.1175/1520-0450(1982)021<1594:AHRMOT>2.0.CO;2.