AceCast_Docs

Jan Ising

Aug 02, 2022

ACECAST

1 What is AceCAST? 3

2 How will AceCAST Benefit You? 5

2.1 Index 5

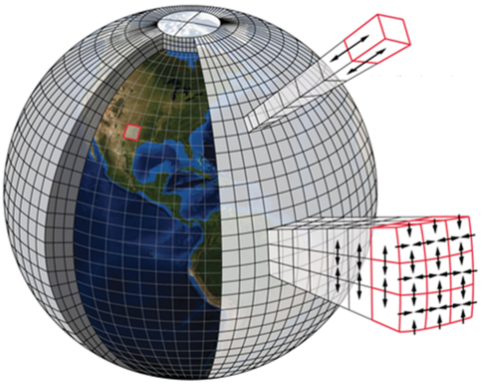


Fig. 1: Numerical weather predictions by splitting the globe into “cubes”. Credit

CHAPTER

ONE

WHAT IS ACECAST?

TempoQuest's (TQI's) software product, **AceCAST, accelerates** the world's most widely used regional weather forecasting model, known as the Weather Research and Forecasting (WRF) model, which is supported by the U.S. National Center for Atmospheric Research (NCAR). **WRF** is widely utilized by more than 36,000 registered users located in 162 countries, and is a next-generation mesoscale **Numerical Weather Prediction** (NWP) system designed to serve both operational forecasting and atmospheric research needs. WRF features multiple dynamical cores and a software architecture allowing for computational parallelism and system extensibility. WRF is suitable for a broad spectrum of applications across scales ranging from meters to thousands of kilometers.

AceCAST is a powerful cutting-edge software **powered by Graphics Processing Units (GPUs)** that enable acceleration of the WRF model. AceCAST is the product of half-a-decade of punctilious research and development that empowers WRF users to secure striking performance optimizations using the superior massive parallelism of GPU hardware versus traditional CPU computation. AceCAST encompasses an ample set of refactored common WRF physics and dynamics modules, and namelist options with NVIDIA CUDA or OpenACC GPU programming techniques, allowing a wide swath of users to adopt AceCAST painlessly as a **drop-in replacement for existing WRF configurations**.

GPUs enable exceptional acceleration by using multi-threaded, many-core processors with tremendous computational speed and very high memory bandwidth. The combined features of general-purpose supercomputing, high parallelism, high memory bandwidth, **low cost**, and compact size are what make a GPU-based system an appealing alternative to a massively parallel system made up of commodity CPU clusters. TempoQuest's software is a result of specialized programming and optimization techniques coupled with a deep understanding of high-performance computing, NWP, atmospheric science, WRF configurations, GPUs, and CUDA code. The TQI developed CUDA-based GPU WRF is currently the world's fastest and highest resolution weather forecasting model.

CHAPTER

TWO

HOW WILL ACECAST BENEFIT YOU?

Weather forecasts have a critical impact on the economy and extreme weather events are one of the greatest risks in the world. High quality weather forecasts require a tremendous amount of computational power, which TempoQuest has addressed by accelerating the WRF model by utilizing GPUs. Compared to standard computing methods, AceCAST is currently the world's fastest and highest resolution weather forecasting model. Running WRF via AceCAST enables users to run forecast and research simulations with accelerated time-to-solution processes at higher resolutions, lower cost, and deeper insight. The capabilities of AceCAST provides meteorologists and end-users the products derived by WRF that deliver greater awareness of localized weather phenomena that global weather models are unable to discern, or which are missed at lower resolutions. The compute performance improvement makes high resolution deterministic and probabilistic forecasts practical at lower cost than running WRF on Central Processing Units (CPUs), and the forecast simulation acceleration **has consistently improved forecast processing speed by a factor of five**.

2.1 Index

2.1.1 Before You Begin

AceCAST is simply a modified implementation of the standard CPU-based WRF model distributed by NCAR (checkout the wrf users page here) that runs on high-performance GPUs. Our goal is to make AceCAST a simple and convenient alternative to running WRF on CPUs and as such have preserved as much of the standard WRF functionality as possible. It is assumed that users are at least somewhat familiar with running the standard WRF model prior to running AceCAST. If you are not familiar with the WRF model we suggest you explore the various resources provided on the WRF users page before continuing.

This guide is intended to help existing WRF users become familiar with the mechanics of running AceCAST and how they can take advantage of AceCAST's performance advantages to enhance their modeling capabilities.

2.1.2 System Requirements

- OS & Host Architecture:
  - This distribution of AceCAST targets generic x86-64 and Power9 Linux systems. Note: Support is not guaranteed for any particular Linux distribution but this release has been tested successfully on a variety of distributions when using the recommended installation methods.

Note

This AceCAST distribution targets generic x86-64 and Power9 Linux systems. Support is not guaranteed for any particular Linux distribution, but this release has been tested successfully on a variety of distributions when using the recommended installation methods.

- CUDA-Capable GPU and CUDA Toolkit:
  - This AceCAST distribution is compatible with GPU compute capabilities 3.5, 6.0, 7.0 and 8.0 and requires a CUDA runtime driver installation compatible with the CUDA 11.0 toolkit (i.e. driver versions >= 450.36.06 – tip: you can find your version by running nvidia-smi). Power9 version is compiled with CUDA 10.1 for compatibility with older system drivers. This should already be installed on most systems with GPUs by default. If it is not, please refer to the documentation for installing the necessary cuda components.

Note

TQI extensively tested the model on Intel (Haswell and Skylake) CPUs with NVIDIA V100 GPUs on a CentOS Linux version 7 platform. The IBM Power9 version is extensively tested on the U.S. Department of Energy Summit supercomputer.

2.1.3 Installation

Prerequisites

Before attempting to install AceCAST users should ensure they have access to a compatible system with NVIDIA GPUs (see *System Requirements*).

Contact and User Support

Please do not hesitate to contact support@tempoquest.com for any issues or questions that you may have during the installation process.

Download AceCAST

To download AceCAST, navigate to the *Releases* page and follow the download instructions for the newest release of AceCAST. Once the AceCAST tarball has been downloaded, unpack it using the **tar** command as follows.

```
[acecast-user@localhost]$ tar -xf AceCASTv1.3+linux.x86_64.tar.gz
```

Package Contents

Once unpacked you should see a new *AceCAST* directory with a file structure similar to this:

```
AceCAST
├── benchmarks
│     - a directory intended to hold AceCAST benchmark test
│       case data
├── README
├── run
│   ├── acecast-advisor.sh
│   │     - script intended to help users determine namelist
│   │       support and computational scaling extents
│   ├── acecast.exe
│   │     - AceCAST GPU-WRF executable -- this is the GPU
│   │       equivalent of wrf.exe
```

(continues on next page)

(continued from previous page)

```
    ├── CCN_ACTIVATE.BIN
    ├── diffwrfs
    │     - utility for comparing WRF/AceCAST input and
    │       output files
    ├── GENPARM.TBL
    ├── gpu-launch.sh
    │     - MPI wrapper script for assigning GPUs to each
    │       MPI process
    ├── LANDUSE.TBL
    ├── ndown.exe
    │     - precompiled ndown.exe executable (see WRF
    │       documentation) -- runs on CPU
    ├── ozone.formatted
    ├── ozone_lat.formatted
    ├── ozone_plev.formatted
    ├── README.namelist
    ├── real.exe
    │     - precompiled real.exe executable -- runs on CPU
    ├── RRTMG_LW_DATA
    ├── RRTMG_SW_DATA
    ├── SOILPARM.TBL
    ├── VEGPARM.TBL
    └── wrf_stats_ck
          - utility script for getting basic performance
            metrics of AceCAST/WRF runs
└── scripts
    └── install_deps.sh
          - script for installing the shared libraries
            that are required to run the various
            executables in this package
```

Many of these files should already look familiar to seasoned WRF users. We have added short annotations above describing files and directories that we believe are worth noting.

Acquire a License

AceCAST is a licensed software package and as such requires a valid license to run. A 60-day trial license can be acquired by registering at the TempoQuest Registration Page. After registering you should recieve an email containing your trial license. We suggest placing this file in the *AceCAST/run* directory. If your 60-day trial has ended please contact support@tempoquest.com to request an extension and/or a quote.

Running the Installation Script

To ensure compatibility with the precompiled executables in the AceCAST distribution it is highly recommend to use the dependency installation script to install the various compilers and shared libraries required at runtime (found at *AceCAST/scripts/install_deps.sh*). This script installs the following components:

- NVIDIA 20.7 (formerly PGI) compiler suite (includes NVIDIA-compiled, CUDA-aware OpenMPI build)
- HDF5
- NetCDF

- Parallel NetCDF
- Generates a script that ensures the proper runtime environment for running AceCAST

```
Usage:  ./install_deps.sh [--install-secondary-packages-rpm] [--install-secondary-
↪packages-deb]
```

This will prompt the user to specify an installation directory for these dependencies (default: *$HOME/tqi-build/20.7*). It will also generate a script (default: *$HOME/tqi-build/20.7/env.sh*) that should be sourced to setup your runtime environment correctly prior to running any of the executables contained in the AceCAST distribution (*i.e. real.exe, acecast.exe, wrf.exe, diffwrfs, etc.*).

Important: This process can take up to an hour to complete and requires ~16GB of storage.

Example:

```
[acecast-user@localhost]$ cd AceCAST/scripts/
[acecast-user@localhost]$ ./install_deps.sh
Welcome to the AceCAST Package Installer
Checking for supported architecture
Supported architecture detected arch=Linux_x86_64
Enter directory you would like to install NVIDIA HPC SDK and associated packages in:
/home/ec2-user/tqi-build/20.7
Installing AceCAST dependent packages in /home/ec2-user/tqi-build/20.7
Installing NVIDIA HPC SDK version 20.7
Downloading file - https://developer.download.nvidia.com/hpc-sdk/20.7/nvhpc_2020_207_
↪Linux_x86_64_cuda_multi.tar.gz
Extracting archive - nvhpc_2020_207_Linux_x86_64_cuda_multi.tar.gz
.................................successfully extracted archive - nvhpc_2020_207_Linux_
↪x86_64_cuda_multi.tar.gz
Successfully patched SDK
Success: Successfully installed NVIDIA HPC SDK version 20.7
Building HDF5
Downloading file - http://www.hdfgroup.org/ftp/HDF5/releases/hdf5-1.12/hdf5-1.12.0/src/
↪hdf5-1.12.0.tar.bz2
Extracting archive - hdf5-1.12.0.tar.bz2
.successfully extracted archive - hdf5-1.12.0.tar.bz2
Success: Successfully installed HDF5 version 1.12.0 in /home/ec2-user/tqi-build/20.7/hdf5
Building NetCDF C Libraries
Downloading file - https://github.com/Unidata/netcdf-c/archive/v4.7.4.tar.gz
Extracting archive - v4.7.4.tar.gz
.successfully extracted archive - v4.7.4.tar.gz
Success: Successfully installed NETCDF-C version 4.7.4 in /home/ec2-user/tqi-build/20.7/
↪netcdf
Building NetCDF Fortran Libraries
Downloading file - https://github.com/Unidata/netcdf-fortran/archive/v4.5.3.tar.gz
Extracting archive - v4.5.3.tar.gz
successfully extracted archive - v4.5.3.tar.gz
Success: Successfully installed NETCDF-Fortran version 4.5.3 in /home/ec2-user/tqi-build/
↪20.7/netcdf
Building Parallel NetCDF Libraries
Downloading file - https://parallel-netcdf.github.io/Release/pnetcdf-1.12.1.tar.gz
Extracting archive - pnetcdf-1.12.1.tar.gz
```

(continues on next page)

(continued from previous page)

```
successfully extracted archive - pnetcdf-1.12.1.tar.gz
Success: Successfully installed parallel-netcdf version 1.12.1 in /home/ec2-user/tqi-
↪build/20.7/pnetcdf
Success: Successfully Installed AceCAST Dependency Packages
Environment setup script generated at /home/ec2-user/tqi-build/20.7/env.sh
```

We suggest running without the optional flags *–install-secondary-packages-rpm* or *–install-secondary-packages-deb* initially. If the script reports any issues then continue to the *Installing with Secondary Dependencies* section. Otherwise you can continue to the *Running AceCAST* page.

Installing with Secondary Dependencies

To run successfully, the installation script when run as-is (i.e. without the *–install-secondary-packages-rpm* or *–install-secondary-packages-deb* flags) requires a number of common packages that provide various utilities and libraries. Since they are so common it isn't unusual for these packages to already be available on many users' systems. When this is not the case these packages need to be installed with package repository managers such as *yum* or *apt-get* depending on the OS is running. Notably, this also requires administrator privileges.

If you would like to indicate that these packages should be installed, first ensure you have sudo privilages to install packages with *yum* or *apt-get* then run the *install_deps.sh* script with the appropriate flag:

```
Usage for RPM-based Linux Distributions:      ./install_deps.sh --install-secondary-
↪packages-rpm
Usage for Debian-based Linux Distributions:   ./install_deps.sh --install-secondary-
↪packages-deb
```

Troubleshooting

If you are still having issues, please check out the *Troubleshooting* section.

2.1.4 Running AceCAST

Before attempting to run make sure you have installed AceCAST properly and acquired a valid license to use the software (see *Installation Guide*).

Input Data

The first step in any AceCAST/WRF workflow is to generate input data for the model. AceCAST intentionally uses the same exact *namelist.input, wrfbdy*, wrfinput*, etc.* files that are used by the standard CPU-WRF model. The only restrictions are that they must be compatible with *WRF version 3.8.1* and the namelist options must be supported by AceCAST (see *Generating Input Data* and *Creating A Namelist*). Although you can use your own test data, to keep things simple for this exercise it is highly recommended to start with the input data from one of our *benchmark test cases*. In this example we will use the *Easter500 benchmark*, which is a good test case for a small number of GPUs and for convenience we download and unpack the benchmark data in the *AceCAST/benchmarks* directory:

```
[acecast-user@localhost]$ cd AceCAST/benchmarks/
[acecast-user@localhost]$ wget https://tqi-public.s3.us-east-2.amazonaws.com/datasets/
↪easter500.tar.gz
...
```

(continues on next page)

(continued from previous page)

```
[acecast-user@localhost]$ tar -xf easter500.tar.gz
[acecast-user@localhost]$ ls
easter500  easter500.tar.gz
[acecast-user@localhost]$ tree easter500
easter500
├── namelist.input
├── namelist.wps
├── wrfbdy_d01
└── wrfinput_d01
```

Setting Up the Simulation Run Directory

Although you are free to run your simulations in the *AceCAST/run* directory, we suggest you create a new directory and link/copy any files required at runtime to that directory. This includes the following:

- AceCAST executable (*acecast.exe*)
- AceCAST license (something like *acecast-trial.lic*)
- MPI wrapper script (*gpu_launch.sh*)
- static runtime data files (*CCN_ACTIVATE.BIN, GENPARM.TBL, LANDUSE.TBL, etc.*)
- namelist file (*namelist.input*)
- input data (*wrfbdy*, wrfinput*, etc.*)

Tip: We consider it best practice to create a new directory for each simulation you run. This can help you avoid common mistakes when running large numbers of simulations and also allows you to run multiple simulations simultaneously if you have the compute resources to do so.

For our example we will be running with 4 GPUs on a single compute node (*localhost*). Given this, we create an appropriately named simulation run directory at *AceCAST/test/easter500-4GPU* and link the necessary runtime files into this directory:

```
[acecast-user@localhost]$ cd AceCAST
[acecast-user@localhost]$ mkdir -p test/easter500-4GPU
[acecast-user@localhost]$ cd test/easter500-4GPU/
[acecast-user@localhost]$ ln -s ../../run/* .
[acecast-user@localhost]$ ln -s ../../benchmarks/easter500/* .
```

At this point your *easter500-4GPU* directory contents should look something like this:

```
easter500-4GPU
├── acecast-advisor.sh -> ../../run/acecast-advisor.sh
├── acecast.exe -> ../../run/acecast.exe
├── acecast-trial.lic -> ../../run/acecast-trial.lic
├── CCN_ACTIVATE.BIN -> ../../run/CCN_ACTIVATE.BIN
├── diffwrfs -> ../../run/diffwrfs
├── GENPARM.TBL -> ../../run/GENPARM.TBL
├── gpu-launch.sh -> ../../run/gpu-launch.sh
├── LANDUSE.TBL -> ../../run/LANDUSE.TBL
├── namelist.input -> ../../benchmarks/easter500/namelist.input
```

(continues on next page)

(continued from previous page)

```
├── namelist.wps -> ../../benchmarks/easter500/namelist.wps
├── ndown.exe -> ../../run/ndown.exe
├── ozone.formatted -> ../../run/ozone.formatted
├── ozone_lat.formatted -> ../../run/ozone_lat.formatted
├── ozone_plev.formatted -> ../../run/ozone_plev.formatted
├── README.namelist -> ../../run/README.namelist
├── real.exe -> ../../run/real.exe
├── RRTMG_LW_DATA -> ../../run/RRTMG_LW_DATA
├── RRTMG_SW_DATA -> ../../run/RRTMG_SW_DATA
├── SOILPARM.TBL -> ../../run/SOILPARM.TBL
├── VEGPARM.TBL -> ../../run/VEGPARM.TBL
├── wrfbdy_d01 -> ../../benchmarks/easter500/wrfbdy_d01
├── wrf.exe -> ../../run/wrf.exe
└── wrfinput_d01 -> ../../benchmarks/easter500/wrfinput_d01
```

Setting Up Your Runtime Environment

Prior to running AceCAST, we need to setup the runtime environment by sourcing the *~/tqi-build/20.7/env.sh* script that was generated by the *install_deps.sh* script during the *Installation*:

```
[acecast-user@localhost]$ source ~/tqi-build/20.7/env.sh
```

Note: If you installed the AceCAST dependencies in a non-default location, the env.sh script will be located in the directory you specified during the installation.

This modifies your *PATH* and *LD_LIBRARY_PATH* variables so that *acecast.exe* can properly link with the shared libraries for NetCDF, HDF5, etc..

Launching AceCAST with MPI

AceCAST uses MPI to enable it to run on multiple GPUs just like WRF does (when compiled for *dmpar*) to run on multiple CPU cores. The standard AceCAST distribution uses an OpenMPI build that is included with the NVIDIA HPC SDK installation (see *Installation*) and typically use the associated *mpirun* launcher to run *acecast.exe*.

Note: In some cases the NVIDIA HPC SDK build of OpenMPI may not be compatible with your system. If you run into any MPI-related issues or poor multi-GPU performance, please contact support@tempoquest.com to discuss alternative builds or other solutions.

General AceCAST usage can be summarized as follows:

```
Usage:  mpirun [MPIRUN_OPTIONS] ./gpu-launch.sh ./acecase.exe
```

We always recommend that you use one MPI task per each GPU you intend to run on. This is accomplished through the proper choice of *MPIRUN_OPTIONS* as well as the *gpu-launch.sh* MPI wrapper script. The goal of the former is to launch the correct number of MPI tasks on each node. The *gpu-launch.sh* script (note that this is run by each MPI task independently) then sets the *ACC_DEVICE_NUM* environment variable (see NVHPC Environment Variables) for each task to ensure the one-to-one mapping of GPUs to their respective tasks. For the majority of users the *gpu-launch.sh* can be used as-is but there are some cases where this may need to be modified (example: running 4 simulations

simultaneously each on their own GPU on a single node), in which case users can find more information in *Modifying the gpu-launch.sh Script*.

> **Warning:** Currently, AceCAST doesn't prevent you from running with multiple MPI tasks per GPU, which can degrade performance as well as cause significant GPU memory limitations. It is important to make sure you are using a single GPU per MPI task.

Note that although the multi-node usage can vary significantly from system to system, the single node use case can nearly always be generalized to:

```
Single Node Usage: mpirun -n <NUM_GPUS> ./gpu-launch.sh ./acecast.exe
```

For our example we are run with 4 GPUs on a single node and can therefore follow this single node usage pattern.

```
[acecast-user@localhost]$ mpirun -n 4 ./gpu-launch.sh ./acecast.exe
 starting wrf task             0  of             4
 starting wrf task             1  of             4
 starting wrf task             2  of             4
 starting wrf task             3  of             4
```

If the run was successful, you should see a message stating *SUCCESS COMPLETE WRF* near the end of the *rsl.error.0000* file.

```
[acecast-user@localhost]$ tail rsl.error.0000
Timing for main: time 2020-04-12_23:59:12 on domain   1:    0.13889 elapsed seconds
Timing for main: time 2020-04-12_23:59:24 on domain   1:    0.13829 elapsed seconds
Timing for main: time 2020-04-12_23:59:36 on domain   1:    0.13934 elapsed seconds
Timing for main: time 2020-04-12_23:59:48 on domain   1:    0.13824 elapsed seconds
Timing for main: time 2020-04-13_00:00:00 on domain   1:    0.14919 elapsed seconds
Timing for Writing wrfout_d01_2020-04-13_00_00_00 for domain        1:    1.76981␣
↪elapsed seconds
Timing for Writing restart for domain        1:    7.45465 elapsed seconds
d01 2020-04-13_00:00:00 wrf: SUCCESS COMPLETE WRF
Checking-in/releasing AceCAST Licenses
Successfully checked-in/released AceCAST Licenses.
```

Summary and Next Steps

In this section we covered the basics of running AceCAST through an example where we ran the *Easter500* benchmark test case with 4 GPUs on a single node. By using input data from one of our benchmark test cases, we were able to focus on the fundamental mechanics of running the AceCAST software before moving on to other critical topics such as generating input data and choosing a namelist. These will be covered in the next sections *Generating Input Data* and *Creating A Namelist*.

2.1.5 Generating Input Data

Under Construction

Where to get input data

Free

1. NCEP
2. UNIDATA THREDDS
3. Amazon Web Services
    - HRRR
4. NOAA THREDDS
5. NOAA Big data
6. University of Utah (HRRR)

Paid

1. ECMWF

Other

1. Air Resource Laboratory

2.1.6 Creating A Namelist

Under Construction

2.1.7 Releases

Version 1.3

Download

To download the AceCAST v1.3 distribution package use the following link:

AceCAST v1.3 for Linux x86-64: AceCASTv1.3+linux.x86_64.tar.gz

Alternatively, if you would like to download the package from the command line you can simply copy the url from above and use a tool such as *wget* or *curl* to download the file. Example:

```
wget https://tqi-public.s3.us-east-2.amazonaws.com/distros/AceCASTv1.3%2Blinux.
↪x86_64.tar.gz
```

Important: AceCAST requires a valid license file to run. Register for a free 60-day trial license by clicking here. Contact support@tempoquest.com for more licensing information.

New Features

The following table summarizes the newly supported namelist options in version 1.3.

Namelist Variable	Supported Options	Description
&fdda (grid and obs nudging)		
grid_fdda	0, **1***	grid-nudging fdda on (=0 off) for each domain
grid_sfdda	0, **1*, 2***	surface fdda switch 0: off; 1: nudging selected surface fields; 2: FASDAS (flux-adjusting surface data assimilation system)
&stoch (Stochastic parameterization schemes)		
skebs	0, **1***	stochastic forcing option: 0=none, 1=SKEBS
&physics		
fractional_seaice	0, **1***	treat sea-ice as fractional field (1) or ice/no-ice flag (0) works for sf_sfclay_physics=1,2,5,or 7. If fractional_seaice = 1, also set seaice_threshold = 0.
rdlai2d	.false., **.true.***	use LAI from input; false means using values from table if sst_update=1, LAI will also be in wrflowinp file
ua_phys	.false., **.true.***	Option to activate UA Noah changes: a different snow-cover physics in Noah, aimed particularly toward improving treatment of snow as it relates to the vegetation canopy. Also uses new columns added in VEGPARM.TBL
iz0tlnd	0, **1***	thermal roughness length for sfclay and myjsfc (0 = old, 1 = veg dependent Chen-Zhang Czil) for mynn sfc (0=Zilitinkevitch,1=Chen-Zhang,2=mod Yang,3=const zt)

*** Indicates newly supported option**

Stochastic Parameterization Schemes

Stochastic parameterization schemes can be used to identify areas of model uncertainty in ensemble simulations by applying small perturbations at every time step to each ensemble member. To apply a stochastic parameterization scheme to your simulation, please reference the WRF User's Guide at:

https://www2.mmm.ucar.edu/wrf/users/docs/user_guide_v4/v4.0/users_guide_chap5.html#stochastic

Grid Nudging

Two computationally inexpensive Four-Dimensional Data Assimilation (FDDA) techniques, Analysis Nudging and Surface Analysis Nudging, are available in this version of AceCAST (Version 1.3). For more information on how to use analysis nudging please refer to the documentation at:

https://www2.mmm.ucar.edu/wrf/users/docs/user_guide_v4/v4.0/users_guide_chap5.html#gridnudge

Improvements and Bug Fixes

Noah LSM (sf_surface_physics = 2) Improvement

The noah lsm scheme was using a lot of thread-level private automatic arrays. This was computationally expensive and also caused memory issues for larger patch sizes. In these cases users would have encountered errors similar to the following prior to the run failing:

```
INITIALIZE THREE Noah LSM RELATED TABLES
 Tile Strategy is not specified. Assuming 1D-Y
WRF TILE   1 IS    139 IE    275 JS    137 JE    272
WRF NUMBER OF TILES =   1
 Tile Strategy is not specified. Assuming 1D-Y
WRF TILE   1 IS    279 IE    556 JS    198 JE    393
WRF NUMBER OF TILES =   1
FATAL ERROR: FORTRAN AUTO ALLOCATION FAILED
FATAL ERROR: FORTRAN AUTO ALLOCATION FAILED
FATAL ERROR: FORTRAN AUTO ALLOCATION FAILED
```

We have reworked the noah lsm components and users should no longer see this issue moving forward.

Adaptive Time Stepping (ADT) Bug Fix

Ever since the release of ADT in version 1.1 we have encountered subtle issues for simulations using ADT with multiple GPUs. This turned out to be an issue where the *num_sound_steps* was miscalculated causing numerical instabilities that resulted in CFL errors and the model eventually blowing up. Example rsl log output:

```
d01 2020-12-09_00:00:00            15  points exceeded cfl=2 in domain␣
↪d01 at time 2020-12-09_00:00:00 hours
d01 2020-12-09_00:00:00  max_vert_cfl=             Inf
Timing for main (dt= 10.00): time 2020-12-09_00:00:10 on domain   1:  ␣
↪21.61837 elapsed seconds
d01 2020-12-09_00:00:10             5  points exceeded cfl=2 in domain␣
```

(continues on next page)

(continued from previous page)

```
↪d01 at time 2020-12-09_00:00:10 hours
d01 2020-12-09_00:00:10  max_vert_cfl=              Inf
Timing for main (dt=  1.00): time 2020-12-09_00:00:11 on domain   1:  ␣
↪ 0.44749 elapsed seconds
Failing in Thread:1
call to cuStreamSynchronize returned error 700: Illegal address during␣
↪kernel execution
```

This rather slippery bug has finally been identified and fixed. We highly encourage users to consider using ADT moving forward.

Dependency Installation Script Improvement

Many AceCAST users do not have administrator access on the systems they install and run AceCAST on. Typically these systems already have the required software packages installed on the machine but we have found that a number of users have reported issues with the libcurl headers and libraries when running the dependency installation script provided with the AceCAST distribution. We have disabled the unnecessary feature of the netcdf-c configuration that required libcurl packages.

Improved Error Messaging

We have found that users were experiencing kernel launch error messages in cases where they were actually running out of GPU memory. This behavior has been corrected.

Modified Default Grid Decomposition Strategy

AceCAST will now default to a 1-dimensional patch decomposition strategy when running on multiple GPUs. This typically improves performance by up to 20% in our experience due to improved MPI buffer packing/unpacking and I/O read/write data access patterns. We therefore decided to make this the default decomposition strategy. Users can still explicitly specify the decomposition using the *nproc_x* and *nproc_y* namelist options if desired.

Known Issues

Easter1500 16x V100 GPU failure

Currently AceCAST encounters an issue when running the Easter1500 benchmark on 16 V100 GPUs with a 4x4 patch decomposition. Due to the new decomposition strategy users are unlikely to encounter this issue. Regardless we would like to make sure that users are aware that it exists.

Version 1.2

- **Nesting**
  - This release includes a large number of new and improved features, the primary of which is nesting. Both 1-way and 2-way nesting is now fully supported with the only notable exceptions being the inability to use vertical nesting and restricting the user to using interp_method_type=2 (sint). If either of these are required for your simulations please contact support@tempoquest.com to ensure that we prioritize their development for future versions.
- **Physics Additions**
  - cu_physics = 1, 11 (Kain-Fritsch and Multi-scale Kain-Fritsch cumulus schemes)

  - cu_rad_feedback = .true. (Sub-grid cloud effect to the optical depth in radiation)
  - kf_edrates = 1 (Add entrainment/detrainment rates and convective timescale output variables)
  - kfeta_trigger = 1, 2, 3 (KF trigger option; cu_physics=1 only)
  - lightning_option = 3 (Lightning parameterization; predicting the potential for lightning activity)
  - iccg_method = 2 (Coarsely prescribed intra-cloud and cloud-to-ground partitioning method)
  - use_mp_re = 1 (see bugfix)
  - scalar_pblmix = 1 (Mix scalar fields consistent with PBL option)
  - grav_settling = 1, 2 (Gravitational settling of fog/cloud droplets)
  - topo_shading = 1 (Neighboring-point shadow effects for solar radiation)
  - slope_rad = 1 (Slope effects for solar radiation)
  - swint_opt = 1 (Interpolation of short-wave radiation based on the updated solar zenith angle between SW call)
  - o3input = 2 (ozone input option for radiation, using CAM ozone data; ozone.formatted)
  - sf_sfclay_physics = 5 (MYNN surface layer scheme)
  - icloud = 2, 3 (cloud effect to the optical depth in radiation)

- **Dynamics Additions**
  - scalar_adv_opt = 2, 3, 4 (advection options for scalar variables)
  - h_sca_adv_order = 6 (6th order horizontal scalar advection)
  - h_mom_adv_order = 6 (6th order horizontal momentum advection)

- **Performance improvements**
  - The initialization overhead time has been an issue for users with short simulations. We have significantly improved the allocation and physics initialization routines (over 10X faster in many cases) to ensure this overhead is nearly negligible when compared to the total runtime for any simulation, regardless of the simulation length.
  - We have addressed an issue where the Kessler scheme (mp_physics = 1) was significantly slower than it should have been. We are now seeing up to 30X speedup for this component and users should be able to confidently use this option.
  - We have significantly optimized the performance of the Noah land-surface scheme (sf_surface_physics = 2), which should give users approximately a 5-10% overall speedup for simulations using this scheme.
  - We have made some universal memory optimizations that have shown up to 10% overall runtime speedups in some cases.

- **Improvements over the base WRF-CPU implementation**
  - There is a bug in the base WRF code (https://github.com/wrf-model/WRF) in all previous releases (currently version 4.2.2) that caused issues when using multi-scale KF (cu_physics=11) on outer nests but not the inner nests (example for 2-domain simulation: cu_physics = 11, 0). This is a common configuration for nested runs since the inner nests may run at convection-resolving resolutions but the coarse domains require a cumulus scheme. This caused the model to produce incorrect results. A bug report has been submitted to the WRF developers but this issue has been resolved in the AceCAST v1.2 release and is currently safe for users running such configurations.
  - The base WRF v3.8.1 code had issues with lightning_option = 3 causing crashes at runtime. This issue has been resolved in AceCAST.

- **Bug Fixes**
  - Fixed issue with adaptive time stepping where the CFL condition was not calculated correctly causing longer time steps that would cause stability issues.
  - Fixed issue where effective radii computed in mp schemes were incorrectly modified by RRTMG.
- **AceCAST Advisor Tool**
  - We have modified both the support-check and scaling-advisor tools to ensure they account for nested runs and implicitly-defined options.
- **Feature Development Targets for Version 1.3**
  - Release v1.3 will incorporate a variety of new features. Our development targets are prioritized by user requests. Please contact support@tempoquest.com if you have any requests for new features. Currently we intend on implementing the following options.
    - Observational Nudging (&fdda namelist options)
    - Fractional Seaice (fractional_seaice = 1 and associated suboptions)
    - Stochastic Parameterization Schemes (&stoch namelist options)
  - Although I/O Quilting is supported in AceCAST to the extent that it is also supported in WRF, there are significant memory limitations that cause the I/O server processes to fail at runtime quite frequently in both AceCAST and WRF. I/O Quilting could have significant benefits for GPU execution with AceCAST if we could make the implementation more reliable. We are currently exploring this opportunity for version 1.3 or later.

Version 1.1.2

- Release 1.1.2 adds beta support for IBM Power9 systems on Linux. This Power9 version is intended for research use only. TQI acknowledges computational resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

Version 1.1.1

- This release does not incorporate any new features. This release incorporates changes necessary to enable counting, floating licenses. We have also cleaned up much of the output from the license checkout/checkin tasks.

Version 1.1

- AceCAST has been compiled and tested with NVIDIA HPC SDK (20.7) and CUDA 11. This version has support for A100 architecture GPUs.
- **Physics Additions**
  - Thompson (mp_physics = 8) & Thompson aerosol-aware microphysics (mp_physics = 28)
  - MYNN surface layer (sf_sfclay_physics = 5)
  - MYNN 3rd level TKE scheme (bl_pbl_physics = 6)
  - RUC land-surface model (sf_surface_physics = 3)
- **Dynamics Additions**
  - do_avgflx_em = 1 (Output time-averaged mass-coupled advective velocities)

  - momentum_adv_opt = 3 (5th-order WENO) advection option
  - moist_adv_opt = 2,3,4 advection options

- **Miscellaneous**
  - Support for adaptive time stepping
    * diag_print = 1 (printing out time series of basic model diagnostics)
    * Performance optimizations for WSM6 (mp_physics = 6), YSU PBL (bl_pbl_physics = 1), and BMJ (cu_physics = 2) schemes

Version 1.0.1

- **Diagnostics**
  - We have ported a significant selection of diagnostics options. The following options are now available to AceCAST.
  - **&time_control:**
    * nwp_diagnostics = 1
    * output_diagnostics = 1
  - **&afwa**
    * afwa_diag_opt = 1
    * afwa_ptype_opt = 1
    * afwa_vil_opt = 1
    * afwa_radar_opt = 1
    * afwa_severe_opt = 1
    * afwa_icing_opt = 1
    * afwa_vis_opt = 1
    * afwa_cloud_opt = 1
    * afwa_therm_opt = 1
    * afwa_buoy_opt = 1
    * afwa_bad_data_check = 1
  - **&diags**
    * p_lev_diags = 1
    * z_lev_diags = 1
    * ! all associated suboptions

Note

The following afwa options are still not available in this version:

1. afwa_turb_opt = 1
2. afwa_hailcast_opt = 1

Please contact support@tempoquest.com if you would like us to consider supporting any other specific diagnostics options in future versions.

- **Physics**
  - We have added support for the following physics options:
    * Mellor-Yamada-Janjic TKE scheme (bl_pbl_physics = 2)
    * Monin-Obukhov (Janjic) scheme (sf_sfclay_physics = 2)

Version 1.0

- **Testing**
  - AceCAST v1.0 has been thoroughly tested at all stages of model development and ready for user evaluation. We rigorously evaluated 12 main physics and majority of dynamics options for numerical and performance aspects using numerous coarse and mesoscale simulations. Additionally, scaling, domain size, boundary, resolution, integration order, and IO sensitivity experiments have been performed to provide a robust high-performance NWP model.
- **Updates**
  - **Licensing Changes**
    * We have moved from providing a generic trial license within the distribution package itself to providing individual trial licenses for each user. The trial licenses will be sent to the user via email after registering at https://tempoquest.com/acecast-registration/. The trial license will be valid for 60 days beginning the day of registration.
  - **Dependency installation script improvements**
    * Added secondary dependency installation functionality for RPM-based and Debian-based Linux distributions using the yum and apt-get package managers. Although this isn't necessary for most users where these secondary deps are already installed, this may be useful on systems that do not have these packages installed. Note that using this option requires sudo access.
    * Usage for RPM-based Linux Distributions: ./install_deps.sh –install-secondary-packages-rpm
    * Usage for Debian-based Linux Distributions: ./install_deps.sh –install-secondary-packages-deb
    * Added checks to ensure each installation step succeeded before moving on to the next. Issues with the dependency installation script will now be much clearer and easier to identify.
    * Improved namelist configuration checks
    * Extended configuration support checks to ensure a valid set of options is chosen at runtime.
  - **AceCAST advisor script**
    * Added a namelist checking utility (run/acecast-advisor.sh), which advises the user how to change a namelist based on what options are supported by AceCAST as well as the number of GPUs one should use when running the given namelist.
  - **Performance Optimizations**
    * Gravity Wave Drag (gwd_opt = 1) - Our initial implementation of the gwd option was rather slow due to a lack of parallelism. This scheme has been reimplemented to exploit the available parallelism and is no longer a significant performance bottleneck.

* RRTMG Longwave Radiation (ra_lw_physics = 4) - The memory overhead of the RRTMG-LW scheme has been significantly reduced, which has reduced allocation times and improved computational performance as well.
* MYNN PBL (bl_pbl_physics = 5) - The MYNN PBL scheme has been reworked to exploit more parallelism.

Version 1.0-beta

- Initial public release of AceCAST
- Supported Platforms
  - This release of AceCAST has a single generic distribution targeting x86-64 Linux systems. Support is not guaranteed for any particular Linux distribution but this release has been tested successfully on a variety of distributions when using the recommended installation methods (see README.ACECAST). This distribution has been built for NVIDIA GPU compute capabilities 3.5, 6.0 and 7.0. We extensively tested the model on Intel (Haswell and Skylake) CPUs with NVIDIA V100 GPUs on a CentOS Linux version 7 platform.

2.1.8 Benchmarks

Under Construction

Easter100

Input Files: easter100.tar.gz

Easter500

Input Files: easter500.tar.gz

Easter1500

Input Files: easter1500.tar.gz

2.1.9 Advanced Topics

Under Construction

Modifying the gpu-launch.sh Script

2.1.10 Troubleshooting

1. General
   - Please ensure that the installation script created the environment script (env.sh):

     ```
     $ cd ../tqi-build/20.7
     ```

   - Once in this directory, type:

```
$ ls
```

- You should see an env.sh script in this directory. If you see this script and the message in the terminal, then AceCAST was installed successfully.

2. The install_deps.sh script keeps failing

Provided below is a list of required library dependencies for AceCAST. Please check to see if these libraries have already been installed on your system. Please use the dependency installation script to ensure compatibility with the AceCAST binary executable:

- NVIDIA HPC SDK (formerly PGI) compiler suite which includes NVIDIA-compiled, CUDA-aware OpenMPI build (2020, Version 20.7)
- HDF5 (Version 1.12.0)
- NETCDF-C (Version 4.7.4)
- NETCDF-Fortran (Version 4.5.3)
- parallel-netcdf (Version 1.12.1)

Even if you have these libraries and versions already installed on your system, you will still have to run the dependency installation script because this script not only installs the required library dependencies, but it also generates a script that ensures a proper runtime environment for running AceCAST (env.sh).

Typically, most linux-based systems already have these optional packages pre-installed but you will also find it useful to have:

- yum
- apt-get
- git
- tar
- wget
- gcc
- make
- which
- etc…

Library dependency check

Run the following commands listed below to see if your system already has the required library versions and ensure that the required library versions were installed properly on your system.

1. Check to see if HDF5 (Version 1.12.0) is installed. Enter the following in the command line and press enter:

```
$ h5dump -version
```

- This command tells you the HDF5 version that is installed on your system and should return:

```
$ h5dump: Version 1.12.0
```

2. Check to see if netCDF-C (Version 4.7.4) is installed. Enter the following in the command line and press enter:

```
$ nc-config -version
```

- This command tells you the netCDF-C version that is installed on your system and should return:

```
$ netCDF 4.7.4
```

3. Check to see if netCDF-Fortran (Version 4.5.3) is installed. Enter the following in the command line and press enter:

```
$ nf-config -version
```

- This command tells you the netCDF-Fortran version that is installed on your system and should return:

```
$ netCDF-Fortran 4.5.3
```

4. Check to see if NVIDIA (Version 20.7) is installed. Enter the following in the command line and press enter:

```
$ pgfortran -V
```

- This command tells you the NVIDIA compiler version that is installed on your system and should return:

```
$ pgfortran (aka nvfortran) 20.7-0 LLVM 64-bit target on x86-64
  Linux -tp haswell
  PGI Compilers and Tools
  Copyright (c) 2020, NVIDIA CORPORATION.  All rights reserved.
```

5. Check to see if PnetCDF (Version 1.12.1) is installed. Enter the following in the command line and press enter:

```
$ pnetcdf_version
```

- This command tells you the PnetCDF version that is installed on your system and should return:

```
$ PnetCDF Version:        1.12.1
```

- If you are using Power9, please issue the following commands before running the dependency installation script:

```
$ module purge
$ export TPFLAGS=-tp=pwr9
```

Don't see your issue addressed? Let's have a *discussion* about it!

2.1.11 About

Under Construction

WSV3 is the visualization program that pairs with AceCAST. To learn more click here

A quick overview of the capabilities of WSV3

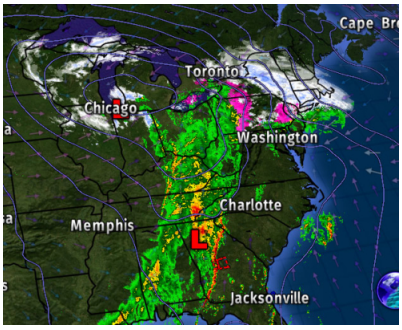
Noteworthy features include but are not limited to:

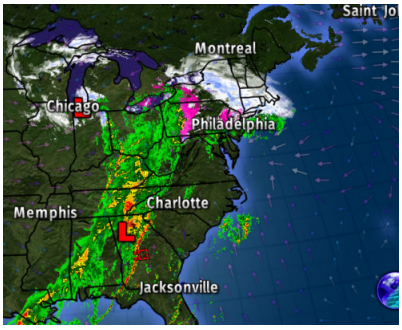
1. Quality private-server-backed National Radar mosaic - 3 options, based on MRMS
2. NEXRAD Level 2 radar data including proprietary LiveScan and LiveComposite technologies
    - Derived Level 2 radial products (Velocity Dealiasing, SRV, MEHS, ET, Max Downdraft Gust, etc.)
    - Support for multiple public internet data sources for redundancy in high-demand events
3. NEXRAD Level 3 radar data including visual depicts of non-radial point products (storm tracks, hail icons, MESO/TVS, etc.)
4. Extensive suite of high-temporal-resolution GOES-16 satellite imagery
    - Two RGB multispectral visualizations included (True Color RGB + IR/VIS Sandwich)
    - 5 bands of CMIP imagery for GOES-16 CONUS
    - Add-on subscription adds access to full 16-band output of G16/17 Full Disk, CONUS, and MESO rapid-scan 1-minute imagery with multiple RGB multispectral products
5. Severe Weather tracking abilities
    - Point/path extrapolation from NWS warning objects and manual tracks
    - Tracking information from Level 3 radar derived storm cells
    - Rollover ETA storm arrival information and city time-to-impact list
6. NWS Severe Watche outlines and Severe Warning polygons
7. MRMS data
8. Mesoanalysis fields
9. SPC convective outlooks
10. LSR (Local Storm Report) icons
11. METAR + NDBC surface observations
12. WPC Surface Analysis
13. GOES-R GLM lightning data with add-on subscription
14. Extensive foreecast model data system
    - Defaults include all major NCEP models (GFS, NAM, HRRR, RAP, HIRESW)
    - 48-Hour HRRR
    - Customizable to add any GRIB2-format internet model data; pulls from NCEP-format HTTP servers
15. Tropical data layers
16. Winter weather data including WPC snow/ice accumulation probabilities
17. Highly customizable GIS/Vector rendering engine and background/terrain rendering engine for unique, brandable map appearance
18. Import custom Placefile data/ESRI shapefiles
19. Spatial tools for distance measurement to aid forecasting and motion extrapolation

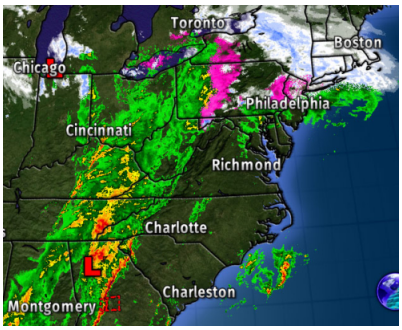
See the videos below for more details:

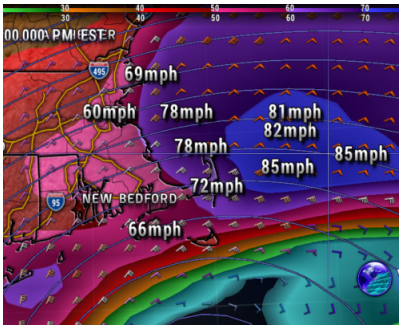
2.1.12 Gallery

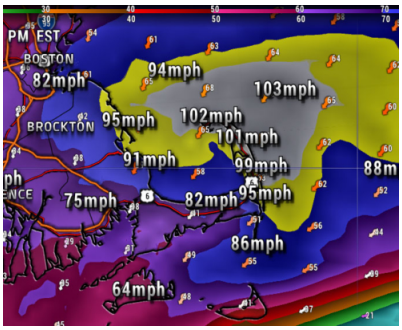
Under Construction



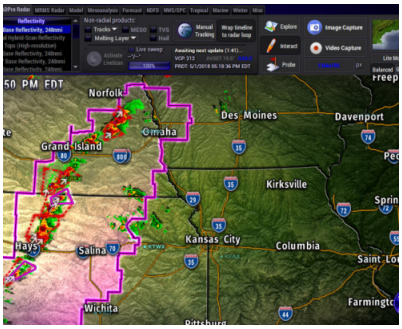


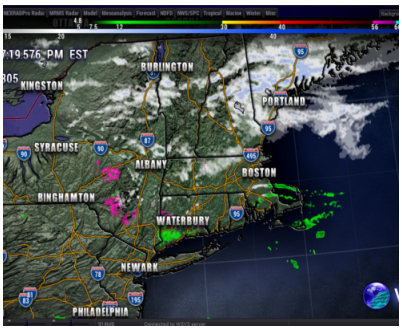


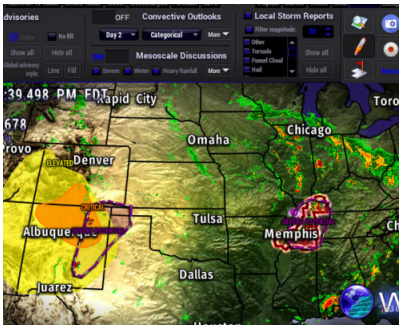

69mph
60mph
78mph
81mph
82mph
78mph
85mph
85mph
72mph
NEW BEDFORD
66mph


BOSTON
82mph
94mph
103mph
BROCKTON
95mph
102mph
101mph
91mph
99mph
75mph
82mph
95mph
86mph
64mph

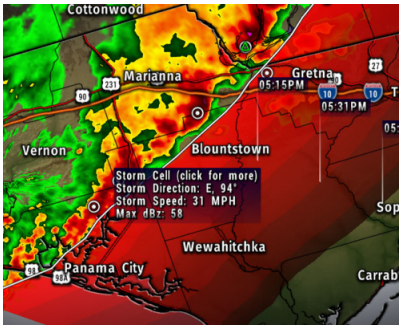

Storm ETA
6:03 PM EDT
40:45 until impact
Monroe
Arcadia
Jonesboro
Natchitoches
Jena
Natchez
Vicksburg
Yazoo City

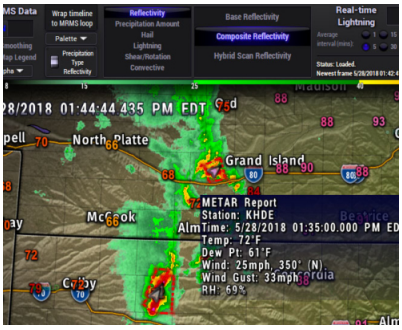

Norfolk
Omaha
Des Moines
Davenport
Grand Island
Kirksville
Kansas City
Columbia
Salina
Wichita


KINGSTON
BURLINGTON
PORTLAND
SYRACUSE
ALBANY
BOSTON
BINGHAMTON
WATERBURY
NEWARK
PHILADELPHIA


Convective Outlooks
Local Storm Reports
Mesoscale Discussions
Rapid City
Chicago
Omaha
Denver
Tulsa
Memphis
Dallas
Juarez


Memphis
Dallas
Jackson
Houston
New Orleans
Acuna
Monterrey


Cottonwood
Marianna
Gretna
05:15PM
05:31PM
Vernon
Blountstown
Storm Cell (click for more)
Storm Direction: E, 94°
Storm Speed: 31 MPH
Max dBz: 58
Wewahitchka
Panama City


Reflectivity
Base Reflectivity
Composite Reflectivity
Real-time Lightning
North Platte
Grand Island
METAR Report
Station: KHDE
Temp: 72°F
Dew Pt: 61°F
Wind: 25mph, 350° (N)
Wind Gust: 33mph
RH: 69%


Kansas City
Bowling Green
Oklahoma City
Memphis
Rome
Dallas
Monroe
Montgomery
New Orleans
Houston

2.1.13 Tools

- Namelist Support table

Variable Names	Input Options	Description
&domains		*dimensions, nesting, parameters*
feedback	0	No feedback from nested domain to the parent domain. Also known
	1	Feedback from nested domain to the parent domain. Also known as t
smooth_option		smoothing option for parent domain; used only with feedback=1
	0	Off
	1	1-2-1 smoothing option for parent domain; used only with feedback=
	2	(default) smoothing-desmoothing option for parent domain; used onl
interp_method_type	2	(default) SINT Horizontal interpolation option
&physics		
Microphysics		(mp_physics)
	1	Kessler scheme
	6	WSM6 scheme
	8	Thompson scheme
	28	Thompson aerosol-aware scheme
Radiation		(ra_lw_physics and ra_sw_physics)
	4	RRTMG LW and RRTMG SW schemes
PBL		(bl_pbl_physics) boundary layer option
	1	YSU scheme; use sf_sfclay_physics = 1
	2	Mellor-Yamada-Janjic TKE Scheme; use sf_sfclay_physics = 2
	5	MYNN 2.5 level TKE scheme; use sf_sfclay_physics = 1, 2, or 5
	6	MYNN 3rd level TKE scheme; use sf_sfclay_physics = 5
Land Surface		(sf_surface_physics) land-surface option (set this before running real
	1	Thermal Diffusion scheme
	2	Unified Noah land-surface model; Includes Noah LSM mosaic sub-o
	3	RUC land-surface model (num_soil_layers = 6)
Surface Layer		(sf_sfclay_physics) surface layer option. The same value should be u
	1	Revised MM5 Monin-Obukhov scheme (Jimenez, renamed in v3.6)
	2	Monin-Obukhov (Janjic Eta) scheme
	5	MYNN
Cumulus		(cu_physics) Cumulus parameterization option
	0	Off (On when dx/dy >10km; Off when dx/dy <4km)
	1	Kain-Fritsch (new Eta) scheme
	2	Betts-Miller-Janjic scheme
	11	Multi-scale Kain-Fritsch scheme (when 4km < dx/dy < 10km)
Miscellaneous		
Gravity wave drag parameterization	1	gravity wave drag option; use when grid size > 10 km (default is 0 =
GRIMS Shallow Cumulus option	3	GRIMS scheme; independent shallow cumulus option (not tied to de
1D mixed layer ocean model	1	Activate a simple ocean mixed layer (oml) model; Only works with s
Time-varying sea-surface temperature	1	real.exe will create wrflowinp file(s) at the same time interval as the a
Radar reflectivity calculation	1	allows radar reflectivity to be computed using mp-scheme specific pa
Lightning potential prediction	3	Predicting the potential for lightning activity (based on Yair et al., 20
&dynamics		*Diffusion, damping options,advection options*
	Majority supported	

Note: Most options can be turned off with a value of 0, where applicable.

Benchmarks

Once AceCAST has been downloaded, there will be a folder called benchmarks in ../AceCASTv1.2/run directory.

- It is recommended to use the test cases provided in the benchmarks directory for getting started with AceCAST.
  - **easter100:**
    - 2km horizontal resolution, 100x100x51 grid, 2 days simulation initialized with HRRR
    - Very small domain intended for quick validation testing.
    - Note that this case is only for testing and should not be used for performance benchmarking due to its small domain size.
  - **easter500:**
    - 2km horizontal resolution, 500x500x51 grid, 2 days simulation initialized with HRRR
    - Larger domain intended for benchmarking on small numbers of GPUs (ex. 1-4 V100 GPUs)
  - **easter1500 (not included by default):**
    - 1km horizontal resolution, 1500x1500x51 grid, 1 hour simulation initialized with HRRR
    - Large, high resolution domain intended for performance benchmarking on larger numbers of GPUs (ex. 8-32 V100 GPUs)
    - You must ensure minimum 11GB space to simulate this case
    - This case is not included in the distribution package
    - download here

AceCAST Scaling Advisor

- The AceCAST Advisor script provides various tools to assist AceCAST users with configuring and running their AceCAST simulations. The primary functionalities are selected with the –tool flag and are summarized below. This should be the primary tool for users to transition their namelists to take advantage of AceCAST.This is the primary tool for users to check and refine their namelist settings to ensure they are compatible with AceCAST. If your namelist settings are not supported by AceCAST, this tool will tell you which namelist setting(s) is/are not compatible, and will provide you with options that are supported by AceCAST. Additionally, this tool also assists users with determining the minimum and maximum number of GPUs that should be used to run their simulation in an optimal amount of time.
- To see more information about the AceCAST Advisor Script (acecast-advisor.sh), make sure you are in the ../AceCASTv1.2/run directory and enter the following in the command line and press enter:

```
$ ./acecast-advisor.sh -help
```

- Invoking this command will list all of the AceCAST Advisor Script options
- usage: acecast-advisor.sh [OPTIONS]
  - **OPTIONS:**
    - -t, –tool <support-check|scaling-advisor> [Default: none]
    - -n, –namelist-file <FILE> [Default: ./namelist.input]
    - -v, –verbose
    - -h, –help

- Tools
    1. Namelist Support Check
        - The namelist support checking tool is used to ensure that all options in a given namelist are supported by AceCAST. The script will tell the user which options are not supported and what supported alternatives the user should consider to adapt their namelist for running with AceCAST.
    2. Scaling Advisor
        - The scaling advisor tool is used to give AceCAST users a general sense of how many GPUs they should use to run their simulations. The scaling characteristics of AceCAST are relatively complex making this a challenging task otherwise. This tool will determine a minimum and maximum number of GPUs that the user should consider initially for a given namelist and GPU hardware specification.
- **Examples:**
    1. ./acecast-advisor.sh -t support-check -namelist-file namelist.input
    2. ./acecast-advisor.sh -tool support-check -namelist-file namelist.input
    3. ./acecast-advisor.sh -t scaling-advisor -namelist-file namelist.input
    4. ./acecast-advisor.sh -tool scale-advisor -namelist-file namelist.input

Note

When using the scaling advisor tool the script should be run on the intended runtime machine and runtime environment. Otherwise the script will not be able to determine critical information about the user's GPU specifications.

2.1.14 Tutorials

Coming Soon!

2.1.15 External Links

- Tempoquest homepage
- WRF homepage
- CPU WRF on-demand
- WSV3 data Visualizer
- Namelist.wps best practices
- Namelist.input best practices

2.1.16 Frequenty Asked Questions

General

1. Q: How much does AceCast Cost?
    - A: Please get in touch with *sales*.
2. Q: What is ACECAST?

- A: ACECAST is like WRF but with GPU's instead of CPU's. This means that you can have higher quality results in a faster time!

3. Q: What kind of runtime improvements can I expect to see?
   - A: This varies, please get in touch with *sales* for specific details.

Runtime

1. Q: I downloaded AceCast, now what?
   - A: Check out the *installation page* to get started!
2. Q: What are the system requirements?
   - A: Refer to *System Requirements*
3. Q: Do I need any other software to run AceCast?
   - A: There are some dependecies required. Please refer to *Installation*

Technical

1. Q: My output is all NaN's, help!
   - A: Check your rsl.error log, if you see CFL errors then decrease your time step in your namelist.input file (~5-6x dx)
2. Q: Why do I get an error when I run install_deps.sh?
   - A: Your system may not be compatible/properly setup, please *get in touch with us* for help.
3. Q: My license has expired what do I do?
   - A: Send us an *email!*.

Other

1. Q: What are the recommended settings to use?
   - A: This varies but generally you can check out these links for WRF namelist.wps and namelist.input best practices.
2. Q: Where can I learn more?
   - A: Read through through the different tabs on the side of this page and then *get in touch with us*

2.1.17 Support

Under Construction

Please do not hesitate to contact support@tempoquest.com for any issues or questions that you may have.

2.1.18 License Information

- After registering to use AceCAST , you should have received an email with information on downloading AceCAST as well as a license file (ex. acecast-trial.lic). This file will be checked by the acecast.exe executable at runtime to ensure the user has a valid license.
- To ensure that AceCAST can evaluate your license at runtime there are *two* options:
  1. Copy the license file to the run directory you are executing acecast.exe in
  2. Point the RLM_LICENSE environment variable to the location of the license file:

```
$ export RLM_LICENSE=$HOME/AceCAST/run/acecast-trial.lic
```

- When running acecast, a successful license checkout should generate output similar to this in the rsl.out.0000 and rsl.error.0000 files:

```
Checking out AceCAST Licenses.
RLM license checkout:
------------------------
Product: acecast
Version: 1
Count:   4
------------------------
Successfully checked out licenses.
```