AceCast Docs

Jan Ising

Nov 22, 2023

ACECAST

CHAPTER

ONE

QUICK LINKS: DOWNLOAD | ONDEMAND | WSV3 | TEMPOQUEST

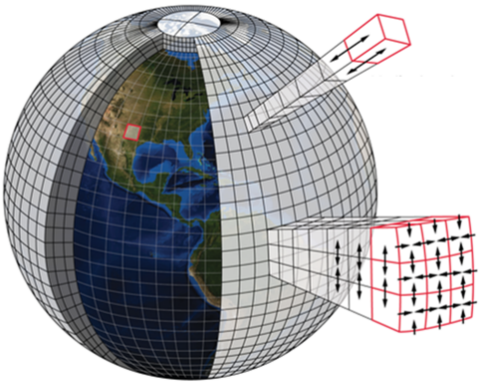


Fig. 1: Numerical weather predictions by splitting the globe into "cubes". Credit

1.1 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.

1.2 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**.

1.2.1 Index

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.

Installation Guide

This is a step-by-step guide to installing AceCAST and its software dependencies on your system locally.

Platform Requirements

Check for compatible OS and CPU architecture

AceCAST is highly integrated with the NVIDIA HPC SDK and is supported on any platforms supported by the NVIDIA HPC SDK (for more details see NVHPC platform requirements). Currently we only provide AceCAST distribution packages (see *Releases*) for Linux x86_64 systems but if you are interested in AceCAST for Linux OpenPOWER or Linux ARM, please contact us at support@tempoquest.com. AceCAST is not supported on Windows machines but note that AceCAST can be run within Linux VMs running on Windows or as a container (see *Containers*).

CPU Architecture:

command

```
uname -m
```

example output

```
x86_64
```

Operating System Info:

command

```
cat /etc/*release
```

example output

```
NAME="Red Hat Enterprise Linux"
VERSION="8.3 (Ootpa)"
ID="rhel"
ID_LIKE="fedora"
VERSION_ID="8.3"
PLATFORM_ID="platform:el8"
PRETTY_NAME="Red Hat Enterprise Linux 8.3 (Ootpa)"
ANSI_COLOR="0;31"
CPE_NAME="cpe:/o:redhat:enterprise_linux:8.3:GA"
HOME_URL="https://www.redhat.com/"
BUG_REPORT_URL="https://bugzilla.redhat.com/"

REDHAT_BUGZILLA_PRODUCT="Red Hat Enterprise Linux 8"
REDHAT_BUGZILLA_PRODUCT_VERSION=8.3
REDHAT_SUPPORT_PRODUCT="Red Hat Enterprise Linux"
REDHAT_SUPPORT_PRODUCT_VERSION="8.3"
Red Hat Enterprise Linux release 8.3 (Ootpa)
Red Hat Enterprise Linux release 8.3 (Ootpa)
```

Make sure your OS/CPU architecture combination is supported by the NVIDIA HPC SDK (see NVHPC platform requirements).

Check for CUDA-capable GPUS

AceCAST can only be run on CUDA-capable GPUs with a compute capability of 3.5 or above. To check for CUDA-capable GPUs run the following:

Checking for GPUs:

command

```
lspci | grep -i nvidia
```

example output

```
01:00.0 3D controller: NVIDIA Corporation GA100 [A100 SXM4 40GB] (rev a1)
41:00.0 3D controller: NVIDIA Corporation GA100 [A100 SXM4 40GB] (rev a1)
81:00.0 3D controller: NVIDIA Corporation GA100 [A100 SXM4 40GB] (rev a1)
c1:00.0 3D controller: NVIDIA Corporation GA100 [A100 SXM4 40GB] (rev a1)
```

Once you have determined what type of GPUs you have you can verify they have a valid compute capability here.

Installing the NVIDIA CUDA Driver

Prior to istalling the NVIDIA HPC SDK you will need to make sure the proper CUDA driver (version 11.0 or higher) is installed. To check if the CUDA driver is already installed on your system you can try running the following:

NVIDIA SMI Command:

command

```
nvidia-smi
```

example output

```
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 470.57.02    Driver Version: 470.57.02    CUDA Version: 11.4     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  NVIDIA A100-SXM...  On   | 00000000:01:00.0 Off |                    0 |
| N/A   28C    P0    50W / 400W |      0MiB / 40536MiB |      0%      Default |
|                               |                      |             Disabled |
+-------------------------------+----------------------+----------------------+
|   1  NVIDIA A100-SXM...  On   | 00000000:41:00.0 Off |                    0 |
| N/A   26C    P0    48W / 400W |      0MiB / 40536MiB |      0%      Default |
|                               |                      |             Disabled |
+-------------------------------+----------------------+----------------------+
|   2  NVIDIA A100-SXM...  On   | 00000000:81:00.0 Off |                    0 |
| N/A   28C    P0    50W / 400W |      0MiB / 40536MiB |      0%      Default |
|                               |                      |             Disabled |
+-------------------------------+----------------------+----------------------+
|   3  NVIDIA A100-SXM...  On   | 00000000:C1:00.0 Off |                    0 |
| N/A   26C    P0    48W / 400W |      0MiB / 40536MiB |      0%      Default |
|                               |                      |             Disabled |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

If the command runs without any issues the CUDA drivers are already installed. Make sure the cuda version is at least 11.0 or higher in the *nvidia-smi* output. If this is not the case then follow the CUDA Installation Guide before moving on. Note that this step requires root access to install.

Installing the NVIDIA HPC SDK

AceCAST requires installation of the NVIDIA HPC SDK version 21.9. You can either follow the NVHPC Installation Guide (make sure to use the archived downloads page at NVHPC 21.9 Downloads) or you can try our quick method below:

NVHPC v21.9 Quick Install:

Quick Installation

```
export NVHPC_INSTALL_DIR=$HOME/nvhpc     # feel free to change this path
export NVHPC_INSTALL_TYPE=single
export NVHPC_SILENT=true
wget https://developer.download.nvidia.com/hpc-sdk/21.9/nvhpc_2021_219_Linux_x86_64_cuda_
↪multi.tar.gz
tar xpzf nvhpc_2021_219_Linux_x86_64_cuda_multi.tar.gz
nvhpc_2021_219_Linux_x86_64_cuda_multi/install

echo '#!/bin/bash'"
export NVARCH=\`uname -s\`_\`uname -m\`
export NVCOMPILERS=$NVHPC_INSTALL_DIR
export MANPATH=\$MANPATH:\$NVCOMPILERS/\$NVARCH/21.9/compilers/man
export PATH=\$NVCOMPILERS/\$NVARCH/21.9/compilers/bin:\$PATH
export LD_LIBRARY_PATH=\$NVCOMPILERS/\$NVARCH/21.9/compilers/lib:\$LD_LIBRARY_PATH
export LD_LIBRARY_PATH=\$NVCOMPILERS/\$NVARCH/21.9/cuda/11.0/lib64:\$LD_LIBRARY_PATH
export LD_LIBRARY_PATH=\$NVCOMPILERS/\$NVARCH/21.9/math_libs/11.0/lib64:\$LD_LIBRARY_PATH

export PATH=\$NVCOMPILERS/\$NVARCH/21.9/comm_libs/mpi/bin:\$PATH
export LD_LIBRARY_PATH=\$NVCOMPILERS/\$NVARCH/21.9/comm_libs/mpi/lib:\$LD_LIBRARY_PATH
export MANPATH=\$MANPATH:\$NVCOMPILERS/\$NVARCH/21.9/comm_libs/mpi/man
" > $NVHPC_INSTALL_DIR/acecast_env.sh
```

Note: This step can take a while depending on your internet speeds. The installation itself typically takes 10 minuts or so.

Updating Environment Script

Note: AceCAST v3.2.2 introduced changes that require updated paths in the environment. To ensure AceCAST v3.2.2 and later link properly at runtime, users who set up the *acecast_env.sh* script prior to v3.2.2 with the Quick Installation commands should use this to update their acecast environment script.

```
export NVHPC_INSTALL_DIR=$HOME/nvhpc     # make sure this is set to what it was when you␣
↪ran the quick install

echo '#!/bin/bash'"
export NVARCH=\`uname -s\`_\`uname -m\`
export NVCOMPILERS=$NVHPC_INSTALL_DIR
export MANPATH=\$MANPATH:\$NVCOMPILERS/\$NVARCH/21.9/compilers/man
export PATH=\$NVCOMPILERS/\$NVARCH/21.9/compilers/bin:\$PATH
export LD_LIBRARY_PATH=\$NVCOMPILERS/\$NVARCH/21.9/compilers/lib:\$LD_LIBRARY_PATH
export LD_LIBRARY_PATH=\$NVCOMPILERS/\$NVARCH/21.9/cuda/11.0/lib64:\$LD_LIBRARY_PATH
```

(continues on next page)

(continued from previous page)

```
export LD_LIBRARY_PATH=\$NVCOMPILERS/\$NVARCH/21.9/math_libs/11.0/lib64:\$LD_LIBRARY_PATH

export PATH=\$NVCOMPILERS/\$NVARCH/21.9/comm_libs/mpi/bin:\$PATH
export LD_LIBRARY_PATH=\$NVCOMPILERS/\$NVARCH/21.9/comm_libs/mpi/lib:\$LD_LIBRARY_PATH
export MANPATH=\$MANPATH:\$NVCOMPILERS/\$NVARCH/21.9/comm_libs/mpi/man
" > $NVHPC_INSTALL_DIR/acecast_env.sh
```

Environment Setup

Notice that a new script is created at *$NVHPC_INSTALL_DIR/acecast_env.sh*. You will need to source this script to setup your environment prior to running AceCAST. Example:

```
source $HOME/nvhpc/acecast_env.sh
```

Installing AceCAST

Download AceCAST Distribution Package

To install AceCAST itself, navigate to the *Version 3.2.2* and copy the download url for AceCAST. You can then download and unpack the distribution using the *wget* and *tar* commands as follows:

```
wget https://tqi-public.s3.us-east-2.amazonaws.com/distros/acecast-v3.2.2%2Blinux.x86_64.
↪haswell.tar.gz
tar -xf acecast-v3.2.2+linux.x86_64.haswell.tar.gz
```

If successful you should see a new directory *acecast-v3.2.2*. The directory structure should look like the following:

```
acecast-v3.2.2
├── acecast
│   └── run
│       ├── acecast.exe
│       ├── ideal.exe
│       ├── ndown.exe
│       ├── real.exe
│       └── tc.exe
├── upp
│   └── exec
│       └── unipost.exe
└── wps
    ├── geogrid.exe
    ├── metgrid.exe
    └── ungrib.exe
```

Note: You should see more files/directories than what is shown here. We are only showing a subset here to give users a sense of the package contents.

Notice that we have added UPP and WPS packages for your convenience since they are frequently used within AceCAST/WRF workflows.

Verify Runtime Environment

One quick way to verify that you have installed and set up your environment correctly in the previous steps is to print the shared libraries used by the *acecast.exe* executable with the *ldd* command.

command

```
ldd acecast-v3.2.2/acecast/run/acecast.exe
```

successful output example

```
linux-vdso.so.1 (0x0000155555551000)
libstdc++.so.6 => /lib64/libstdc++.so.6 (0x0000155554f96000)
libutil.so.1 => /lib64/libutil.so.1 (0x0000155554d92000)
libz.so.1 => /lib64/libz.so.1 (0x0000155554b7b000)
libm.so.6 => /lib64/libm.so.6 (0x00001555547f9000)
libmpi_usempif08.so.40 => /home/samm.tempoquest/nvhpc/Linux_x86_64/21.9/comm_libs/mpi/
↪lib/libmpi_usempif08.so.40 (0x00001555545d0000)
libmpi_usempi_ignore_tkr.so.40 => /home/samm.tempoquest/nvhpc/Linux_x86_64/21.9/comm_
↪libs/mpi/lib/libmpi_usempi_ignore_tkr.so.40 (0x00001555543cb000)
libmpi_mpifh.so.40 => /home/samm.tempoquest/nvhpc/Linux_x86_64/21.9/comm_libs/mpi/lib/
↪libmpi_mpifh.so.40 (0x000015555417e000)
libmpi.so.40 => /home/samm.tempoquest/nvhpc/Linux_x86_64/21.9/comm_libs/mpi/lib/libmpi.
↪so.40 (0x0000155553d3f000)
libdl.so.2 => /lib64/libdl.so.2 (0x0000155553b3b000)
libpthread.so.0 => /lib64/libpthread.so.0 (0x000015555391b000)
librt.so.1 => /lib64/librt.so.1 (0x0000155553713000)
libc.so.6 => /lib64/libc.so.6 (0x0000155553350000)
/lib64/ld-linux-x86-64.so.2 (0x000015555532b000)
libgcc_s.so.1 => /lib64/libgcc_s.so.1 (0x0000155553138000)
libopen-rte.so.40 => /home/samm.tempoquest/nvhpc/Linux_x86_64/21.9/comm_libs/mpi/lib/
↪libopen-rte.so.40 (0x0000155552df4000)
libopen-pal.so.40 => /home/samm.tempoquest/nvhpc/Linux_x86_64/21.9/comm_libs/mpi/lib/
↪libopen-pal.so.40 (0x000015555292b000)
librdmacm.so.1 => /usr/lib64/librdmacm.so.1 (0x0000155552710000)
libibverbs.so.1 => /usr/lib64/libibverbs.so.1 (0x00001555524f1000)
libnuma.so.1 => /usr/lib64/libnuma.so.1 (0x00001555522e5000)
libnvf.so => /home/samm.tempoquest/nvhpc/Linux_x86_64/21.9/comm_libs/mpi/lib/../../../../
↪compilers/lib/libnvf.so (0x0000155551cb0000)
libnvhpcatm.so => /home/samm.tempoquest/nvhpc/Linux_x86_64/21.9/comm_libs/mpi/lib/../../.
↪./../compilers/lib/libnvhpcatm.so (0x0000155551aa5000)
libatomic.so.1 => /usr/lib64/libatomic.so.1 (0x000015555189d000)
libnvcpumath.so => /home/samm.tempoquest/nvhpc/Linux_x86_64/21.9/comm_libs/mpi/lib/../../
↪../../compilers/lib/libnvcpumath.so (0x0000155551468000)
libnvc.so => /home/samm.tempoquest/nvhpc/Linux_x86_64/21.9/comm_libs/mpi/lib/../../../../
↪compilers/lib/libnvc.so (0x0000155551210000)
libnl-3.so.200 => /usr/lib64/libnl-3.so.200 (0x0000155550fed000)
libnl-route-3.so.200 => /usr/lib64/libnl-route-3.so.200 (0x0000155550d67000)
```

As you can see above all of the required libraries were found and are in the expected locations (for example notice the the mpi library libmpi.so.40 was found within the OpenMPI installation of the NVIDIA HPC SDK).

problematic output example

```
linux-vdso.so.1 (0x0000155555551000)
libstdc++.so.6 => /lib64/libstdc++.so.6 (0x0000155554f96000)
libutil.so.1 => /lib64/libutil.so.1 (0x0000155554d92000)
libz.so.1 => /lib64/libz.so.1 (0x0000155554b7b000)
libm.so.6 => /lib64/libm.so.6 (0x00001555547f9000)
libmpi_usempif08.so.40 => not found
libmpi_usempi_ignore_tkr.so.40 => not found
libmpi_mpifh.so.40 => not found
libmpi.so.40 => not found
libdl.so.2 => /lib64/libdl.so.2 (0x00001555545f5000)
libpthread.so.0 => /lib64/libpthread.so.0 (0x00001555543d5000)
librt.so.1 => /lib64/librt.so.1 (0x00001555541cd000)
libc.so.6 => /lib64/libc.so.6 (0x0000155553e0a000)
/lib64/ld-linux-x86-64.so.2 (0x000015555532b000)
libgcc_s.so.1 => /lib64/libgcc_s.so.1 (0x0000155553bf2000)
```

Notice here that the various required MPI libraries (*libmpi.so.40, libmpi_mpifh.so.40, etc.*) were not found. In this case either the NVIDIA HPC SDK was not installed correctly, or the environment was not set up correctly (i.e. the *acecast_env.sh* script was not created or not sourced).

Note: The *ldd* command doesn't guarantee that AceCAST will run correctly but it can be extremely helpful in identifying a number of common issues that users run into regularly.

Acquire A License

AceCAST is a licensed software package and as such requires a valid license to run. A 30-day trial license can be acquired by registering at the AceCAST Registration Page. After registering you should recieve an email containing your trial license (*acecast-trial.lic*). We suggest placing this file in the *acecast-v3.2.2/acecast/run* directory. If your 30-day trial has ended please contact support@tempoquest.com to request an extension and/or a quote.

Running AceCAST

This guide will demonstrate how to run AceCAST by walking you through an example with the *Easter500 benchmark* test case.

Before attempting to run, make sure AceCAST and its dependencies have been installed correctly and that you have a valid AceCAST license to use the software (see *Installation Guide*). Also make sure that you have a valid license file (see *Acquire A License*) and have placed it in your *acecast-v3.2.2/acecast/run/* directory.

For this example we will assume your AceCAST installation is in your home directory (i.e. at *~/acecast-v3.2.2*). If it is somewhere else you will need to modify the code examples accordingly.

Input Data

The first step in any AceCAST/WRF workflow is to generate input data for the model. AceCAST uses the same *namelist.input, wrfbdy*, wrfinput*, etc.* files that are used by the standard CPU-WRF model. The only restriction is that the specified namelist options must be supported by AceCAST (see *Namelist Configuration*). In this guide we will use we will use the *Easter500 benchmark* from our standard *Benchmarks*, which is typically a good test case for a small number of GPUs.

Download Easter500 Test Case Data:

```
cd ~/acecast-v3.2.2/acecast
mkdir benchmarks
cd benchmarks
wget https://tqi-public.s3.us-east-2.amazonaws.com/datasets/v2/easter500.tar.gz
tar -xf easter500.tar.gz
```

At this point your acecast directory should look like this:

```
~/acecast-v3.2.2/acecast
├── benchmarks
│   ├── easter500
│   │   ├── met_em.d01.2020-04-12_00:00:00.nc
│   │   ├── << more met_em.* files >>
│   │   ├── met_em.d01.2020-04-13_12:00:00.nc
│   │   └── namelist.input
│   └── easter500.tar.gz
└── run
    ├── acecast-advisor.sh
    ├── acecast-trial.lic
    ├── acecast.exe
    ├── gpu-launch.sh
    ├── real.exe
    └── << various static data files >>
```

Setting Up the Simulation Run Directory

Next we would like to create a new directory to run the simulation in containing all of the necessary runtime and input files. This will typically include the following:

- **AceCAST Run Files (always found in *~/acecast-v3.2.2/acecast/run/*):**
  - executables (*real.exe, acecast.exe, etc.*)
  - acecast advisor script (*acecast-advisor.sh*)
  - license file (*acecast.lic* or *acecast-trial.lic*)
  - MPI wrapper script (*gpu_launch.sh*)
  - static data files (*CCN_ACTIVATE.BIN, GENPARM.TBL, LANDUSE.TBL, etc.*)
- **Simulation Specific Input Data and Configuration Files (found in *~/acecast-v3.2.2/acecast/benchmarks/easter500* for this example):**
  - namelist file (*namelist.input*)
  - *real.exe* or *acecast.exe* input data (*met_em** or *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 using 4 GPUs and will set up this simulation run directory at *~/acecast-v3.2.2/acecast/easter500-4GPU*:

```
# Create and cd to new run directory
mkdir ~/acecast-v3.2.2/acecast/easter500-4GPU
cd ~/acecast-v3.2.2/acecast/easter500-4GPU

# Link static acecast run files
ln -s ../run/* .

# Link input data files
ln -s ../benchmarks/easter500/met_em.* .

# Copy the namelist file
cp ../benchmarks/easter500/namelist.input .
```

Tip: We typically copy the namelist.input file rather than create a symbolic link like we do with all of the other files here. Since the namelist is modified regularly it is best to make changes to the local copy of the file rather than the original, which can cause confusing problems if the namelist is linked and edited in multiple run directories.

Verify Namelist Configuration

At this point we can use the *acecast-advisor.sh* script to verify that all of the options specified in the namelist are supported by AceCAST. We have an entire section of the documentation dedicated to this topic (see *Namelist Configuration*) but we will keep things simple for this example.

Note: The *Easter500 benchmark* is distributed with a fully supported namelist but we recommend trying out the *acecast-advisor.sh* tool anyways to get a sense of how it works for when you start using your own namelists rather than the one that we provide for this example.

AceCAST Advisor – Support Check Tool

command

```
# cd to the simulation run directory if you aren't already there
./acecast-advisor.sh --tool support-check
```

output for supported namelist

```
********************************************************************************************
*      ___           _____           _      ___      _       _                              *
*     / _ \         /  __ \         | |    / _ \    | |     (_)                             *
*    / /_\ \ ___ ___| /  \/ __ _ ___| |_  / /_\ \ __| |_   ___ ___  ___  _____             *
*    |  _  |/ __/ _ \ |    / _` / __| __| |  _  |/ _` \ \ / / / __|/ _ \|  __|            *
*    | | | | (_|  __/ \__/\ (_| \__ \ |_  | | | | (_| |\ V /| \__ \ (_) | |               *
```

(continues on next page)

(continued from previous page)

```
*    \_| |_/\___\___|\____/\__,_|___/\__| \_| |_/\__,_| \_/ |_|___/\___/|_|       *
*                                                                                 *
***********************************************************************************


WARNING: Namelist file not specified by user. Using default namelist file path: /home/
↪samm.tempoquest/acecast-v3.2.2/acecast/easter500-4GPU/namelist.input

Support Check Configuration:
    Namelist                    : /home/samm.tempoquest/acecast-v3.2.2/acecast/easter500-
↪4GPU/namelist.input
    AceCAST Version             : 3.2.2
    WRF Compatibility Version   : 4.4.2


NOTE: Namelist options may be determined implicitly if not specified in the given␣
↪namelist.

Support Check Tool Success: No unsupported options found -- Ok to use namelist for␣
↪AceCAST execution.
```

output for unsupported namelist

```
***********************************************************************************
*      ___            _____           _      ___      _       _                   *
*     / _ \         /  __ \         | |    / _ \    | |     (_)                  *
*    / /_\ \ ___ ___| /  \/ __ _ ___| |_  / /_\ \ __| |_   ___ ___  ___  ____    *
*    |  _  |/ __/ _ \ |    / _` / __| __| |  _  |/ _` \ \ / / / __|/ _ \|  __|   *
*    | | | | (_|  __/ \__/\ (_| \__ \ |_  | | | | (_| |\ V /| \__ \ (_) | |      *
*    \_| |_/\___\___|\____/\__,_|___/\__| \_| |_/\__,_| \_/ |_|___/\___/|_|      *
*                                                                                 *
***********************************************************************************


WARNING: Namelist file not specified by user. Using default namelist file path: /home/
↪samm.tempoquest/acecast-v3.2.2/acecast/easter500-4GPU/namelist.input

Support Check Configuration:
    Namelist                    : /home/samm.tempoquest/acecast-v3.2.2/acecast/easter500-
↪4GPU/namelist.input
    AceCAST Version             : 3.2.2
    WRF Compatibility Version   : 4.4.2


NOTE: Namelist options may be determined implicitly if not specified in the given␣
↪namelist.

SUPPORT CHECK FAILURE:
    Unsupported option selected for namelist variable mp_physics in &physics: mp_
↪physics=10
    Supported options for namelist variable mp_physics: 0,1,6,8,28
```

(continues on next page)

(continued from previous page)

```
SUPPORT CHECK FAILURE:
    Unsupported option selected for namelist variable cu_physics in &physics: cu_
↪physics=16
    Supported options for namelist variable cu_physics: 0,1,2,11

Support Check Tool Failure: One or more options found that are not supported by AceCAST.␣
↪Please modify your namelist selections based on the previous "SUPPORT CHECK FAILURE"␣
↪messages and run this check again.
```

Setting Up Your Environment

Prior to running the executables in the following sections you will need to make sure your environment is set up correctly as described in the *Installation Guide* (see *Environment Setup*).

Modify OpenMPI Settings (Optional)

The NVIDIA HPC SDK uses an older version of OpenMPI (version 3.1.5). This version is performant and works well on a variety of systems but it can produce some confusing warnings when running MPI jobs. These warnings can be suppressed by setting the *btl_base_warn_component_unused=0* option using the following commands.

```
mkdir -p ~/.openmpi
echo "btl_base_warn_component_unused = 0" > ~/.openmpi/mca-params.conf
```

Note that this only needs to be done one time on any given system.

Running Real

To generate the *wrfinput*, wrfbdy*, etc.* inputs for AceCAST we need to run Real. This works the same way it does for WRF and this process should be familiar for WRF users.

simple usage

```
# cd to the simulation run directory if you aren't already there
mpirun -n <number of cpu cores> ./real.exe
```

Change the *<number of cpu cores>* to the number of cores you would like to use to run *real.exe*.

general usage

```
# cd to the simulation run directory if you aren't already there
mpirun [MPIRUN_OPTIONS] ./real.exe
```

For more details about the *mpirun* command check out the OpenMPI mpirun documentation or try:

```
mpirun --help
```

Note: The *mpirun* command options can vary depending on a number of factors including the number of nodes, CPU cores per node or whether you are running under resource managers (e.g., SLURM, Torque, etc.) to name a few.

If *real.exe* ran successfully then you should see that it generated the input files for AceCAST (*wrfinput*, wrfbdy*, etc.*) and you can also check for a successful completion message in the RSL log files:

command

```
tail -n 5 rsl.error.0000
```

example output

```
d01 2020-04-13_12:00:00 forcing artificial silty clay loam at   11 points, out of  15625
d01 2020-04-13_12:00:00 Timing for processing          0 s.
d01 2020-04-13_12:00:00 Timing for output          1 s.
d01 2020-04-13_12:00:00 Timing for loop #   37 =          5 s.
d01 2020-04-13_12:00:00 real_em: SUCCESS COMPLETE REAL_EM INIT
```

Running AceCAST

General AceCAST usage can be summarized as follows:

```
mpirun [MPIRUN_OPTIONS] gpu-launch.sh [--gpu-list GPU_LIST] acecast.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 sets the *ACC_DEVICE_NUM* environment variable (see NVHPC OpenACC Environment Variables) to the specific GPU id for each MPI task prior to launching the *acecast.exe* executable.

Note: For more information about the *gpu-launch.sh* script check out *GPU Mapping with MPI and the GPU Launch Script*.

For our example we can run with 4 GPUs on a single node:

command

```
mpirun -n 4 ./gpu-launch.sh ./acecast.exe
```

example output

```
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 AceCAST ran successfully then you should see that it generated the *wrfout** files. You should also check for a successful completion message in the RSL log files:

command

```
tail -n 5 rsl.error.0000
```

example output

```
Timing for main: time 2020-04-12_00:59:48 on domain   1:    0.09450 elapsed seconds
Timing for main: time 2020-04-12_01:00:00 on domain   1:    0.09443 elapsed seconds
d01 2020-04-12_01:00:00 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 namelist configuration. These will be covered in the next sections *Generating Input Data* and *Namelist Configuration*.

Namelist Configuration

AceCAST vs WRF Overview

As we have discussed previously, AceCAST is simply a GPU-accelerated implementation of the CPU-based WRF model that is widely used by the community. We strive to make AceCAST a simple drop-in replacement for CPU WRF to keep things as simple as possible for our users. This means it uses the same input files (*wrfinput*, wrfbdy*, etc.*) and configuration files (*namelist.input*) and produces the same output files (*wrfout*, wrfrst*, etc.*) as CPU WRF. It will even produce the same results in the output fields (within acceptable floating-point error tolerances) for simulations using the same input and configuration files as CPU WRF.

The one caveat to the statement above is that AceCAST does not support every single namelist option that is available in CPU WRF (see CPU WRF Namelist Options). WRF has a huge number of features but many of these namelist options are not used by the majority of WRF users. To ensure the quality and the maintainability of AceCAST, we prioritize supporting the namelist options that have been requested by our users.

Since AceCAST limits the namelist options that are otherwise available in CPU WRF (of which there are thousands), we have developed a number of protections and tools to make sure it is clear to users which options are available as well as to prevent the user from using namelist options that aren't supported in AceCAST.

General Recommendations for Users

If you are simply looking to see which options are available in AceCAST, we recommend skipping to the *Interactive Namelist Support Table App* subsection below.

If you already have a namelist (from your CPU WRF configurations or elsewhere) and would like to see if the namelist is supported or not, check out the *AceCAST Advisor – Support Check Tool* subsection below.

If you are interested in using AceCAST but the options specified in your namelist aren't currently supported by AceCAST and there aren't any acceptable alternatives, we would greatly appreciate if you could contact us at support@tempoquest.com to discuss the possibility of supporting those options in future releases.

Interactive Namelist Support Table App

For a complete list of supported namelist options available in AceCAST, we highly recommend checking out the *Namelist Support Table App* tool.

Most options can be turned off with a value of 0, where applicable. AceCAST supports the same output as WRF which includes but is not limited to netCDF and GRIB.

AceCAST Advisor – Support Check Tool

The *acecast-advisor.sh* script is provided within the AceCAST distribution packages themselves. Once AceCAST is installed this script (found in the *acecast/run/* sub-directory) can be used to check if the options specified in an existing namelist are supported by AceCAST as follows:

Note: We understand that some users may want to check if their namelists are supported by AceCAST **prior** to installing AceCAST. We are currently working on an online tool that would perform the same tests in a browser to avoid this issue.

---

command

```
# cd to the simulation run directory if you aren't already there
./acecast-advisor.sh --tool support-check
```

output for supported namelist

```
*******************************************************************************************
*      ___            _____           _      ___      _       _                           *
*     / _ \         /  __ \         | |    / _ \    | |     (_)                          *
*    / /_\ \ ___ ___| /  \/ __ _ ___| |_  / /_\ \ __| |_   ___ ___  ___  _____      *
*    |  _  |/ __/ _ \ |    / _` / __| __| |  _  |/ _` \ \ / / / __|/ _ \|  __|     *
*    | | | | (_|  __/ \__/\ (_| \__ \ |_  | | | | (_| |\ V /| \__ \ (_) | |        *
*    \_| |_/\___\___|\____/\__,_|___/\__| \_| |_/\__,_| \_/ |_|___/\___/|_|        *
*                                                                                         *
*******************************************************************************************


WARNING: Namelist file not specified by user. Using default namelist file path: /home/
↪samm.tempoquest/acecast-v3.0.1/acecast/easter500-4GPU/namelist.input

Support Check Configuration:
    Namelist                    : /home/samm.tempoquest/acecast-v3.0.1/acecast/easter500-
↪4GPU/namelist.input
    AceCAST Version             : 3.0.1
    WRF Compatibility Version   : 4.4.2


NOTE: Namelist options may be determined implicitly if not specified in the given␣
↪namelist.

Support Check Tool Success: No unsupported options found -- Ok to use namelist for␣
↪AceCAST execution.
```

output for unsupported namelist

```
*******************************************************************************************
*      ___           _____          _      ___      _       _                             *
*     / _ \         /  __ \         | |    / _ \    | |     (_)                            *
*    / /_\ \ ___ ___| /  \/ __ _ ___| |_  / /_\ \ __| |_   ___ ___  ___  _____     *
*    |  _  |/ __/ _ \ |    / _` / __| __| |  _  |/ _` \ \ / / / __|/ _ \|  __|    *
*    | | | | (_|  __/ \__/\ (_| \__ \ |_  | | | | (_| |\ V /| \__ \ (_) | |       *
*    \_| |_/\___\___|\____/\__,_|___/\__| \_| |_/\__,_| \_/ |_|___/\___/|_|       *
*                                                                                 *
*******************************************************************************************


WARNING: Namelist file not specified by user. Using default namelist file path: /home/
↪samm.tempoquest/acecast-v3.0.1/acecast/easter500-4GPU/namelist.input

Support Check Configuration:
    Namelist                    : /home/samm.tempoquest/acecast-v3.0.1/acecast/easter500-
↪4GPU/namelist.input
    AceCAST Version             : 3.0.1
    WRF Compatibility Version   : 4.4.2


NOTE: Namelist options may be determined implicitly if not specified in the given␣
↪namelist.

SUPPORT CHECK FAILURE:
    Unsupported option selected for namelist variable mp_physics in &physics: mp_
↪physics=10
    Supported options for namelist variable mp_physics: 0,1,6,8,28

SUPPORT CHECK FAILURE:
    Unsupported option selected for namelist variable cu_physics in &physics: cu_
↪physics=16
    Supported options for namelist variable cu_physics: 0,1,2,11

Support Check Tool Failure: One or more options found that are not supported by AceCAST.␣
↪Please modify your namelist selections based on the previous "SUPPORT CHECK FAILURE"␣
↪messages and run this check again.
```

Generating Input Data

Generally, AceCAST, like WRF requires certain parameters to exist in the input data. For a list of these parameters please refer to the WRF Forum. Below we provide a short list of free and paid locations to retrieve input data from.

Where to get input data

Free

1. National Centers for Environmental Prediction (NCEP)
2. Unidata Thematic Real-time Environmental Distributed Data Services (THREDDS)
3. Amazon Web Services (AWS)
    - High-Resolution Rapid Refresh (HRRR) data
4. NOAA Thematic Real-time Environmental Distributed Data Services (THREDDS)
5. NOAA Big Data
6. University of Utah (High-Resolution Rapid Refresh (HRRR))

Paid

1. European Centre for Medium-Range Weather Forecasts (ECMWF)

Other

1. Research Data Archive (RDA)
2. Air Resource Laboratory (ARL)

Releases

Jump to most recent AceCAST version: *Version 3.2.2*

Release and Versioning Methodology

To ensure consistency and improve overall usability, AceCAST uses a pseudo-semantic versioning scheme (see more on semantic versioning at https://semver.org/). Any given release version will have a version number *MAJOR.MINOR.PATCH*, which can be understood as follows:

- **MAJOR**: Changes in the **MAJOR** version indicate an upgrade to a new version of WRF. For example AceCAST versions *1.*.** were all based on WRF version *3.8.1* while all AceCAST versions *2.*.** will be based on WRF version *4.2.2*.
- **MINOR**: Changes in the **MINOR** version indicate added support for new namelist options or other important features. For example AceCAST version *1.3.0* added support for a number of options such as *grid_fdda=1* and *fractional_seaice=1*, neither of which were available in AceCAST version *1.2.0*.
- **PATCH**: Changes in the **PATCH** version indicate a bug fix related to a supported namelist option or feature.

For any given version we will provide release notes and download information within its own subsection on this page.

Tip: Not sure what we mean by **supported namelist options**? Check out the *Namelist Configuration* section.

Version 3.2.2

Skip to *Downloads*

Release Notes

New in v3.2.2

- Added support for observational nudging (obs_nudge_opt = 1) and all associated sub-options. Observational nudging is a method of nudging the model where point near observations are nudged based on model error at the observation site. For more information on observational nudging check out https://www2.mmm.ucar.edu/wrf/users/docs/user_guide_v4/v4.4/users_guide_chap5.html#obsnudge.
- Added support for the *multi_bdy_files = T* option. This option can be used to run the preprocessing components (ungrib, metgrid, real) to generate the AceCAST boundary conditions asynchronously during the execution of the WRF/AceCAST simulation itself. For more information on its usage see WRF Docs - Use of Multiple Lateral Condition Files).
- Added support for the widely used NoahMP land surface option *sf_surface_physics = 4*. This includes support for all of the NoahMP-related sub-options including all valid choices for *dveg*, *opt_crs*, *opt_btr*, *opt_run*, *opt_sfc*, *opt_frz*, *opt_inf*, *opt_rad*, *opt_alb*, *opt_snf*, *opt_tbot*, *opt_stc*, *opt_gla*, *opt_rsf*, *opt_soil*, *opt_pedo*, *opt_crop*, *opt_irr*, *opt_irrm*, *opt_infdv*, *opt_tdrn*, and *noahmp_output* in the *&noah_mp* namelist section. The exceptions to this are that *opt_crop = 2* and *opt_run = 5* are not supported. Note that we made some important improvements to NoahMP that are outlined in the Improvements and Bug Fixes section below.
- Added support for the IEVA (Implicit Explicit Vertical Advection) option *zadvect_implicit = 1*. For grids with large aspect ratios (dx/dz >> 1) that permit explicit convection, the large time step is limited by the strongest updraft that occurs during integration. This results in time step often 20-30% smaller, or requires the use of w-filtering, such as latent-heat tendency limiting. Regions of large vertical velocities are also often very small relative to the domain. The IEVA scheme permits a larger time step by partitioning the vertical transport into an explicit piece, which uses the normal vertical schemes present in WRF, and a implicit piece which uses implicit transport (which is unconditionally stable). The combined scheme permits a larger time step than has been previously been used and reduced w-filtering. (Wicker and Skamarock, 2020, MWR)

Improvements and Bug Fixes

- The NoahMP implementation in WRF v4.2.2 has significant bugs that needed to be addressed (see https://forum.mmm.ucar.edu/threads/strange-t2-bias-in-newer-wrf-versions.12259/). Due to this we have decided to implement the newest NoahMP version for the WRF code which will likely be included in WRF version 4.5.2 or later. This includes bug fixes and a number of significant improvements that are important for anyone using the NoahMP surface layer scheme in AceCAST (*sf_surface_physics* option 4).
- Implemented a tiled version of the MYNN PBL schemes (both bl_pbl_physics = 5 and bl_pbl_physics = 6). The previous implementations had massive GPU memory requirements that were overly restrictive for users of these schemes. The new tiled implementation will dynamically determine tile sizes based on the available GPU memory at runtime.
- Fixed issue in Revised MM5 surface layer scheme (sf_sfclay_physics = 1) where in certain conditions the model would hang due to an infinite loop.
- Revised MM5 will now report an error and forcefully exit if it attempts to use an invalid index into a lookup table during the surface layer calculation. This occurs when the model is becoming unstable and unrealistic values are being passed through the model. Previously, in both WRF and AceCAST, this situation would have caused an

obscure seg-fault with no explanation. We thought it would be helpful instead to report when this issue occurs and stop the simulation afterwards.

- AceCAST now reports where significant CFL violations occur, which can be very useful when users encounter issues with numerical instabilities in their simulations.
- Fixed a bug that was introduced in WRF v4.4.2 in the AFWA diagnostics that caused all of the CAPE-related fields to be zero. These fields are now calculated correctly.
- In version 3.0 we introduced optimizations that improved RRTMG performance. Due to a CUDA compiler bug this ended up causing the model to crash in many cases when running on GPUs with compute capabilities older than 8.0. These performance optimizations have been reverted in this version to ensure portability of the code on older GPUs and will be reintroduced in a future version of AceCAST when the CUDA compiler issues are resolved by NVIDIA.

Known Issues

MYNN PBL Sub-Options

Both the *icloud_bl = 0* and *bl_mynn_cloudpdf = 0* options fail when using the MYNN PBL option (*bl_pbl_physics = 5*). If these options are critical for your simulations please contact us at support@tempoquest.com to ensure that we prioritize fixing this issue.

Downloads

- AceCAST version 3.2.2 for Linux x86-64: AceCASTv3.2.2.tar.gz

---

Important: Check out the *Installation Guide* for further installation instructions.

Tip: If you would like to download the package from the command line you can use the *wget* or *curl* commands with the download link url from above.

Version 3.1.0

Skip to *Downloads*

Release Notes

New in v3.1.0

- Added support for the Purdue-Lin microphysics *mp_physics=2*. This is a sophisticated scheme that has ice, snow and graupel processes, suitable for real-data high-resolution simulations.
- Added support for all AFWA diagnostics options. For more information on these options check out (http://www2.mmm.ucar.edu/wrf/users/docs/AFWA_Diagnostics_in_WRF.pdf).
- Added support for spectral nudging *grid_fdda=2*. See WRF user guide - Analysis Nudging Runs for more information.

- Added support for isotropic diffusion *mix_isotropic=1*.
- Added support for Morrison microphysics *mp_physics=10*. Double-moment ice, snow, rain and graupel for cloud-resolving simulations.
- Added support for the wind turbine drag parameterization scheme *windfarm_opt=1*. It represents sub-grid effects of specified turbines on wind and TKE fields. For more information on using this option see WRF README.windturbine.
- Added support for restart runs *restart=T*.
- Added support for Morrison double-moment microphysics with CESM aerosols *mp_physics = 40*.
- Added support for the *insert_init_cloud = T* option, which turns on estimation of initial model clouds.
- Added support for *ra_call_offset = -1* (calls radiation before output).
- Added support for all user-specified values of the *blend_width* option. The *blend_width* option determines the number of grid points in the terrain blending zone from the coarse grid to the fine grid for nested domains.
- Added support for all aerosol input options to RRTMG *aer_opt=1*, *aer_opt=2* and *aer_opt=3*.
- AceCAST has been modified to enable use within the UEMS forecasting framework. Please contact *support@tempoquest.com* for more information regarding using AceCAST in UEMS.
- AceCAST executables now link to the NVIDIA HPC SDK and CUDA libraries dynamically. Users who have already installed the NVIDIA HPC SDK v21.9 for AceCAST may need to update their environment setup scripts accordingly to ensure the correct libraries are found at runtime (see *Installing the NVIDIA HPC SDK*).

Improvements

- Using the runtime I/O field modifications with the *iofields_filename* option was incredibly slow when users had significant numbers of changes since the associated routines were called on every history interval unnecessarily. This is now done a single time at the start of the simulation removing nearly all overhead associated with this option.

Known Issues

Illegal address during kernel execution in RRTMG

A number of users have reported an issue where AceCAST fails with the following message:

```
WRF TILE   1 IS      1 IE    500 JS      1 JE    500
WRF NUMBER OF TILES =   1
an illegal memory access was encountered in ../UWisc/RRTMG_LW/rrtmg_lwrad_cuda.cu at␣
↪line 698
```

We believe this may be a problem with the CUDA rutime/drivers and are investigating the issue. One thing that may help users in the meantime is to use different values for the RRTMG tile size by setting the *ACECAST_RRTMG_LW_NUM_TILES* environment variable and running again:

```
# Example setting the number of tiles to 3
export ACECAST_RRTMG_LW_NUM_TILES=3
mpirun -n 4 ./gpu-launch.sh ./acecast.exe
```

We suggest trying tile sizes of anything between 1 and 20. In some cases this doesn't fix the issue.

MYNN PBL Sub-Options

Both the *icloud_bl = 0* and *bl_mynn_cloudpdf = 0* options fail when using the MYNN PBL option (*bl_pbl_physics = 5*). If these options are critical for your simulations please contact us at support@tempoquest.com to ensure that we prioritize fixing this issue.

Downloads

- AceCAST version 3.1.0 for Linux x86-64: AceCASTv3.1.0.tar.gz

---

Important: Check out the *Installation Guide* for further installation instructions.

Tip: If you would like to download the package from the command line you can use the *wget* or *curl* commands with the download link url from above.

Version 3.0.1

Skip to *Downloads*

Release Notes

The AceCAST version 3.0.1 release includes major updates to implement the CPU-WRF version 4.4.2 release, which is the newest release of WRF (as of Feb. 18th 2023). For reference, AceCAST version 2 implemented the CPU-WRF version 4.2.2 release. If you would like more information regarding the WRF updates that were implemented in this new version of AceCAST, check out the release notes for WRF versions 4.2.2 through 4.4.2.

In addition, AceCAST version 3.0.1 includes a number of new features and bug fixes that are outlined below.

New in v3.0.1

- Added support for full 3D diffusion option *diff_opt = 2*
- Added support for LES-specific options including *km_opt = 2*, *km_opt = 3* and *m_opt = 1*
- Added support for Rayleigh damping *damp_opt = 2*
- Added support for the "original" scalar advection options *moist_adv_opt = 0*, *chem_adv_opt = 0*, *tracer_adv_opt = 0*, *scalar_adv_opt = 0* and *tke_adv_opt = 0*
- Added support for water and ice friendly aerosols option *wif_input_opt = 1* for use with Thompson aerosol aware microphysics (*mp_physics = 28*)
- Added support for various accumulated diagnostic options including any user-specified values for *bucket_mm*, *bucket_J* and *prec_acc_dt* as well as support for *acc_phy_tend = 1*
- Added support for UA Noah LSM snow-cover physics option *ua_phys = .true.*
- Added support for using no microphysics option *mp_physics = 0*

Improvements

- Performance optimizations for RRTMG shortwave and longwave schemes (*ra_sw_physics = 4* and *ra_lw_physics = 4*) as well as for WSM6 microphysics (*mp_physics = 6*). Although the impact of these optimizations will vary significantly from case to case, these optimizations resulted in overall speedups of up to 15% during our testing.
- Improvements to the performance profiling activated with the environment variable *ACECAST_USE_TIMERS=true*. The top-down profile generated at the end of the rsl log files is extremely useful but can be hard to interpret for anyone other than the developers of AceCAST. This option now outputs a "summary" of the timing profile which should help users understand where the the time is being spent. Example (from rsl.error.0000 file):

```
Summary:
| ------------------------------- | ------------ | --------- |
|              Name               |   Time (s)   |  Time (%) |
| ------------------------------- | ------------ | --------- |
| WRF Total                       |   200.296238 |    100.00 |
|     Initialization              |    46.051199 |     22.99 |
|         Allocate                |     3.210721 |      1.60 |
|         I/O (Read)              |    41.070188 |     20.50 |
|         I/O (Write)             |     0.000000 |      0.00 |
|         HALO/Nesting (MPI)      |     0.136974 |      0.07 |
|         HALO/Nesting (non-MPI)  |     0.021627 |      0.01 |
|         Compute/Other           |     1.611689 |      0.80 |
|     Integration                 |   154.244787 |     77.01 |
|         I/O (Read)              |     0.769853 |      0.38 |
|         I/O (Write)             |    42.757482 |     21.35 |
|         HALO/Nesting (MPI)      |     5.807679 |      2.90 |
|         HALO/Nesting (non-MPI)  |     3.958668 |      1.98 |
|         Compute/Other           |   100.951104 |     50.40 |
|             LW Radiation        |     4.589823 |      2.29 |
|             SW Radiation        |     9.976138 |      4.98 |
|             Surface Layer       |     0.489929 |      0.24 |
|             Land Surface        |     1.183034 |      0.59 |
|             PBL                 |     5.112687 |      2.55 |
|             Cumulus             |     0.000000 |      0.00 |
|             Microphysics        |     9.959394 |      4.97 |
| ------------------------------- | ------------ | --------- |

d01 2019-11-26_19:00:00 wrf: SUCCESS COMPLETE WRF
```

Bug Fixes

- WRF version 4.1.3 included a bug fix related to the single-scattering albedo and asymmetry input parameters in the RRTMG shortwave scheme (see WRF PR#997). This bug fix was not correctly implemented in AceCAST version 2, which was calculating these values the same way that WRF versions 3.5.1 through 4.1.2 were. This resulted in a slight but clear cold bias in areas with clouds when compared to simulations using newer versions of CPU-WRF. This issue has been fixed in this new version of AceCAST.
- Removed support for cloud overlap options *cldovrlp = 3* and *cldovrlp = 4*. It turned out that our GPU implementation was using *cldovrlp = 2* regardless of what the user specified in their namelist.
- A bug has been fixed where the model would hang at the start of a run when users attempted to use I/O quilting.

- A bug has been fixed in Thompson Microphysics (*mp_physics = 8*) where, with rare but specific patch decompositions, AceCAST did not allocate enough memory for some variables, which caused an *Illegal address during kernel execution* error.

Known Issues

YSU PBL Performance

AceCAST version 3.0.1 introduced changes to the YSU PBL scheme (*bl_pbl_physics = 1*) that degraded the performance. This PBL scheme isn't particularly expensive but this performance issue may offset some of the performance improvements from other schemes introduced in this version of AceCAST. This is a widely used option and we intend on addressing the performance in the near future.

Using WRF Restart Files

AceCAST will fail if you attempt to do a restart run using a restart file that was generated using CPU-WRF rather than another AceCAST run. This is a rare situation but users can avoid this issue by setting the *force_use_old_data = .true.* option in the *&time_control* section of the namelist.

MYNN PBL Sub-Options

Both the *icloud_bl = 0* and *bl_mynn_cloudpdf = 0* options fail when using the MYNN PBL option (*bl_pbl_physics = 5*). If these options are critical for your simulations please contact us at support@tempoquest.com to ensure that we prioritize fixing this issue.

Downloads

- AceCAST version 3.0.1 for Linux x86-64: AceCASTv3.0.1.tar.gz

---

Important: Check out the *Installation Guide* for further installation instructions.

Tip: If you would like to download the package from the command line you can use the *wget* or *curl* commands with the download link url from above.

Version 2.1.0

Release Notes

AceCAST version 2.1.0 includes a number of critical bug fixes as well as support for new options.

New in v2.1.0

- Added support for Tiedtke cumulus physics scheme (*cu_physics = 6*). Note that this completes AceCAST's support for all options associated with the *CONUS* physics suite (*physics_suite = 'conus'*).
- Added support for SST Updates (*sst_update = 1*). This option can be critical for longer simulations where sea surface temperatures and a number of other surface fields vary enough that they should be updated throughout the simulation. For more information WRF Docs – SST Update for more information.
- Added environment variable *ACECAST_NPROC_X*, which can be used to control the MPI domain decomposition at runtime. In many cases this option can be used to significantly improve MPI communication patterns in multi-gpu runs and can reduce overall runtimes by up to 15% in our experience internally (we suggest starting with *ACECAST_NPROC_X=1*).
- Added environment variable *ACECAST_ALIGN_OPT_LEVEL*, which can be used to control if memory dimensions should be aligned to improve memory access at the cost of extra memory overhead. Setting *ACECAST_ALIGN_OPT_LEVEL=0* will typically reduce the memory overhead of a simulation by up to 20% but will reduce the performance as well and is only recommended for users that are highly constrained by GPU memory capacity.

Bug Fixes

- AceCAST dynamically determines a tile size when calculating the RRTMG radiation components to reduce the massive memory overhead that they require (see *GPU Memory Utilization Issue*). The tile size was not being calculated correctly, which caused AceCAST to use significantly more memory than was necessary (up to 100% or more in some cases). This issue has been fixed.
- Fixed issue where AceCAST failed when using the *fractional_seaice = 1* option with any surface layer option other than Revised MM5 (*sf_sfclay_physics = 1*).
- Even though it was working as intended, the *acecast-advisor.sh* script was previously printing the incorrect *AceCAST Version* and *WRF Compatibility Version* when using the *support check* tool. It should now print the correct versions.

Downloads

- AceCAST version 2.1.0 for Linux x86-64: AceCASTv2.1.0.tar.gz

Important: Check out the *Installation Guide* for further installation instructions.

Tip: If you would like to download the package from the command line you can use the *wget* or *curl* commands with the download link url from above.

Known Issues

SSA Calculation in RRTMG

WRF version 4.1.3 included a bug fix related to the single-scattering albedo and asymmetry input parameters in the RRTMG shortwave scheme (see WRF PR#997). This bug fix was not correctly implemented in AceCAST version 2, which is calculating these values the same way that WRF versions 3.5.1 through 4.1.2 were. This results in a slight but clear cold bias in areas with clouds when compared to simulations using newer versions of CPU-WRF.

MYNN PBL Sub-Options

Both the *icloud_bl = 0* and *bl_mynn_cloudpdf = 0* options fail when using the MYNN PBL option (*bl_pbl_physics = 5*). If these options are critical for your simulations please contact us at support@tempoquest.com to ensure that we prioritize fixing this issue.

Version 2.0.0

Release Notes

This is the first release of our highly anticipated upgraded version of AceCAST based on WRF version 4.2.2. This involved a massive rework of the entire code base due to the significant changes between WRF versions 3.8.1 and 4.2.2. For a comprehensive list of supported options, check out the *Namelist Support Table App* page.

Downloads

- AceCAST version 2.0.0 for Linux x86-64: AceCASTv2.0.0.tar.gz

---

Important: Check out the *Installation Guide* for further installation instructions.

Tip: If you would like to download the package from the command line you can use the *wget* or *curl* commands with the download link url from above.

Known Issues

SSA Calculation in RRTMG

WRF version 4.1.3 included a bug fix related to the single-scattering albedo and asymmetry input parameters in the RRTMG shortwave scheme (see WRF PR#997). This bug fix was not correctly implemented in AceCAST version 2, which is calculating these values the same way that WRF versions 3.5.1 through 4.1.2 were. This results in a slight but clear cold bias in areas with clouds when compared to simulations using newer versions of CPU-WRF.

GPU Memory Utilization Issue

The RRTMG radiation options (*ra_sw_physics=4*, *ra_lw_physics=4*) require a significant amount of GPU memory that would typically be highly restictive when users are running with large grids. To mitigate this issue we use a *tiled* version of these RRTMG routines, which break down the grid into smaller chunks that fit into the available GPU memory and perform the radiation calculations for each of these chunks sequentially. **Due to a minor integer overflow issue, this dynamic tile size calculation doesn't currently work for larger grid sizes.** This issue does not effect the results of any simulations but does significantly limit the grid sizes that can be used for any given GPU. This issue will be resolved in the new version of AceCAST.

Fractional Seaice Issue

AceCAST fails with the following message when using the *fractional_seaice = 1* option together with the *sf_sfclay_physics = 2* (eta similarity) or *sf_sfclay_physics = 5* (MYNN) surface layer options:

```
-------------- FATAL CALLED ---------------
FATAL CALLED FROM FILE:  module_surface_driver.G  LINE:    4936
error -- routine not yet implemented
-----------------------------------------
```

If you encounter this issue you can turn off the fractional seaice option (*fractional_seaice = 0*) or use it with the *sf_sfclay_physics=1* surface layer option (Revised MM5). This issue will be resolved in the next release of AceCAST.

Incorrect Version Messaging in the AceCAST Advisor Script

There is currently a bug in the *acecast-advisor.sh* script where the *AceCAST Version* is *1.2* rather than *2.0.0* and the *WRF Compatibility Version* is *3.8.1* rather than *4.2.2*. The script works correctly and the incorrect versions in the output can be ignored.

Version 1.3 and Older

Due to the major changes from AceCAST version *1.** to version *2.**, it is best to use the archived acecast-v1 docs version of the documentation.

Benchmarks

It is recommended to use the test cases provided on this page for getting started with AceCAST.

NCAR Standard Benchmark Test Cases

The *CONUS 12km* and *CONUS 2.5km* benchmarks provided by NCAR are widely used for evaluating the performance of WRF on various systems. For more information check out the WRF V4.4 Benchmarks Page.

CONUS 12km

This is a small benchmark inteded for running on a single GPU. It is a 12-hour simulation over the continental United States (CONUS) on a 425x300 grid at a 12km resolution.

- Download (500 MB): conus12km.tar.gz

CONUS 2.5km

This is a large benchmark inteded for running on larger numbers of GPUs. It is a 6-hour simulation over the continental United States (CONUS) on a 1501x1201 grid at a 2.5km resolution.

- Download (2.3 GB): conus2.5km.tar.gz

TQI-Provided Benchmark Test Cases

Easter 100

- 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.
- Download (20 MB): easter100.tar.gz

Easter 500

- 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)
- Download (10 GB): easter500.tar.gz

Easter 1500

- 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)
- Download (2.5 GB): easter1500.tar.gz

Containers

Containers are a useful solution that bundle applications along with any runtime dependencies that they require. Containers have a number of advantages for application portability and deployment.

Important: This guide is intended to be a supplement addressing the specific topic of running AceCAST containers and as such does not explain other important topics already covered elsewhere. Container users should still take a look at the core topics prior to reading this guide. Of particular importance are the *Before You Begin*, *Running AceCAST* and *Namelist Configuration* pages although users intending to use containers exclusively can notably skip over the *Installation Guide*.

Docker

Docker is a widely used container system and is a good option if you intend on running AceCAST on any number of GPUs on a single node. More information regarding docker can be found at Docker Docs.

Prerequisites

Prior to running the AceCAST containers on a GPU-enabled machine you will need to install the docker engine as well as the necessary nvidia drivers for your system.

Getting the AceCAST Docker Image

Our Docker-based AceCAST container images can be found on the AceCAST DockerHub repository. Once you have chosen the specific image you would like to use you can obtain the image with the docker pull command:

```
[acecast-user@gpu-node]$ docker pull tempoquestinc/acecast:3.2.2
3.2.2: Pulling from tempoquestinc/acecast:3.2.2
f70d60810c69: Already exists
545277d80005: Already exists
1e7f98e28850: Already exists
86f4173eb75d: Already exists
5c52a07ee59c: Already exists
3b22fa154f31: Already exists
2cd3bbcb6d62: Already exists
7751aa45d446: Pull complete
646ba5f2329e: Pull complete
83c0d9a9685f: Pull complete
619ac9d050cc: Pull complete
13f006ddd7e1: Pull complete
Digest: sha256:a6b5311c66200fe0b8c3f1b11d8856847f144565c9d0666b366f11ae8c733722
Status: Downloaded newer image for tempoquestinc/acecast:3.2.2
docker.io/tempoquestinc/acecast:3.2.2
```

Check that the image exists with the docker images command, which lists some basic information about your local images:

```
[acecast-user@gpu-node]$ docker images
REPOSITORY                 TAG              IMAGE ID       CREATED        SIZE
tempoquestinc/acecast:3.2.2      3.2.2        6d6882fef187       5 hours ago    17.6GB
```

The AceCAST image contains all of the software necessary for running AceCAST inside the container including the *NVIDIA HPC SDK* and *AceCAST* itself. The AceCAST executables, static runtime data files, scripts, etc. are installed in the */opt/acecast/run/* directory inside the container.

```
[acecast-user@gpu-node]$ docker run --rm tempoquestinc/acecast:3.2.2 ls /opt/acecast/run
acecast.exe
aerosol.formatted
aerosol_lat.formatted
...
gpu-launch.sh
...
real.exe
...
```

Running AceCAST in a Container

General docker run command usage for AceCAST:

```
docker run [OPTIONS] tempoquestinc/acecast:3.2.2 COMMAND
```

Table 1: Common Docker Run Command Options

Option	Description
–gpus gpu-request	GPU devices to add to the container (‘all’ to pass all GPUs)
–rm	Automatically remove the container when it exits
-w string	Working directory inside the container
-v	Mount volumes from the specified container(s)

For a full description of available options check out the docker run command documentation or run *docker run –help* from the command line.

Users are free to use containers to run AceCAST in whatever way they see fit. Note that you can use the host OS to manage your inputs/outputs and use a container simply to run acecast itself. In this example we have already downloaded the *Easter500* test case data as well as our acecast license file into the *inputs/* subdirectory. This current directory will be mounted inside the container using the *-v* option to the *docker run* command so all of the files you see here will be available inside the container when it is running.

```
[acecast-user@gpu-node]$ ls
inputs  run.sh
[acecast-user@gpu-node]$ ls inputs/
acecast.lic met_em.d01.2020-04-11_12:00:00.nc  met_em.d01.2020-04-11_13:00:00.nc ␣
↪namelist.input
```

We also have a script in the current directory that will be run inside the container in the next step:

```
[acecast-user@gpu-node]$ cat run.sh
#!/bin/bash

# Create directory for running acecast in and cd into it
mkdir -p run
cd run
```

(continues on next page)

(continued from previous page)

```
# Link AceCAST executables and runtime data files into the run directory
# Note that these files only exist in the container so they won't be available from the
# host system that is running the container after the script finishes executing
ln -sf /opt/acecast/run/* .

# Link the input files that we made available to the docker container using the -v option
# that we passed to the docker run command
ln -sf ../inputs/* .

# Run real.exe to generate the wrf input files
# Note that the --allow-run-as-root option for the mpirun command is necessary since the
# user will be root inside the container
mpirun -np 4 --allow-run-as-root ./real.exe

# Run acecast.exe
mpirun -np 4 --allow-run-as-root ./gpu-launch.sh ./acecast.exe
```

To run this script inside the container we use the *docker run* command:

```
[acecast-user@gpu-node]$ docker run --gpus all -v `pwd`:`pwd` -w `pwd` --rm␣
↪tempoquestinc/acecast:3.2.2 ./run.sh
 starting wrf task             1  of             4
 starting wrf task             2  of             4
 starting wrf task             3  of             4
 starting wrf task             0  of             4
 starting wrf task             1  of             4
 starting wrf task             2  of             4
 starting wrf task             3  of             4
 starting wrf task             0  of             4
```

After the container finishes executing the script we should see the wrf output files in the *run/* subdirectory on the host system:

```
[acecast-user@gpu-node]$ ls run/wrfout*
wrfout_d01_2020-04-11_12:00:00
```

Other Useful Examples

Running the AceCAST advisor script:

```
[acecast-user@gpu-node]$ ls
namelist.input
[acecast-user@gpu-node]$ docker run -v `pwd`:`pwd` -w `pwd` --rm tempoquestinc/acecast:3.
↪2.2 /opt/acecast/run/acecast-advisor.sh --tool support-check

*******************************************************************************
*      ___           _____          _      ___      _      _                  *
*     / _ \         /  __ \         | |    / _ \    | |     (_)                 *
*    / /_\ \ ___ ___| /  \/ __ _ ___| |_  / /_\ \ __| |_   ___ ___  ___  _____    *
*    |  _  |/ __/ _ \ |    / _` / __| __| |  _  |/ _` \ \ / / / __|/ _ \|  __|    *
*    | | | | (_|  __/ \__/\ (_| \__ \ |_  | | | | (_| |\ V /| \__ \ (_) | |      *
```

(continues on next page)

(continued from previous page)

```
*    \_| |_/\___\___|\____/\__,_|___/\__| \_| |_/\__,_| \_/ |_|___/\___/|_|       *
*                                                                                 *
***********************************************************************************


WARNING: Namelist file not specified by user. Using default namelist file path: /home/
↪samm/test_acecast/namelist.input

Support Check Configuration:
    Namelist                    : /home/samm/test_acecast/namelist.input
    AceCAST Version             : 3.2.2 (build: linux.x86_64.haswell)
    WRF Compatibility Version   : 4.4.2


NOTE: Namelist options may be determined implicitly if not specified in the given␣
↪namelist.

SUPPORT CHECK FAILURE:
    Unsupported option selected for namelist variable ra_lw_physics in &physics: ra_lw_
↪physics=1,1,1
    Supported options for namelist variable ra_lw_physics: 0,4

SUPPORT CHECK FAILURE:
    Unsupported option selected for namelist variable ra_sw_physics in &physics: ra_sw_
↪physics=1,1,1
    Supported options for namelist variable ra_sw_physics: 0,4

Support Check Tool Failure: One or more options found that are not supported by AceCAST.␣
↪Please modify your namelist selections based on the previous "SUPPORT CHECK FAILURE"␣
↪messages and run this check again.
```

Verify that GPUs are available on the container:

```
[acecast-user@gpu-node]$ docker run --gpus all --rm tempoquestinc/acecast:3.2.2 nvidia-
↪smi
Wed Mar 15 18:14:34 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 470.161.03   Driver Version: 470.161.03   CUDA Version: 11.4     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  NVIDIA A100-SXM...  Off  | 00000000:87:00.0 Off |                    0 |
| N/A   31C    P0    54W / 400W |      0MiB / 40536MiB |      0%      Default |
|                               |                      |             Disabled |
+-------------------------------+----------------------+----------------------+
|   1  NVIDIA A100-SXM...  Off  | 00000000:90:00.0 Off |                    0 |
| N/A   31C    P0    52W / 400W |      0MiB / 40536MiB |      0%      Default |
|                               |                      |             Disabled |
+-------------------------------+----------------------+----------------------+
|   2  NVIDIA A100-SXM...  Off  | 00000000:B7:00.0 Off |                    0 |
```

(continues on next page)

(continued from previous page)

```
| N/A   29C    P0    53W / 400W |      0MiB / 40536MiB |      0%      Default |
|                                |                      |             Disabled |
+--------------------------------+----------------------+----------------------+
|   3  NVIDIA A100-SXM...  Off   | 00000000:BD:00.0 Off |                    0 |
| N/A   29C    P0    55W / 400W |      0MiB / 40536MiB |      0%      Default |
|                                |                      |             Disabled |
+--------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

Advanced Topics

GPU Mapping with MPI and the GPU Launch Script

As discussed in the *Running AceCAST* section, *acecast.exe* is launched with MPI using the *gpu-launch.sh* wrapper script. The purpose of the *gpu-launch.sh* script is to assign unique GPUs to each MPI process, which requires setting the environment variable *ACC_DEVICE_NUM* (see NVHPC OpenACC Environment Variables) to the intended GPU device ID for each process independently. This is done by determining the local MPI rank of the current process and assigning a unique GPU to it based on what GPUs are available on that node. All of this is done automatically by launching the *acecast.exe* executable with *mpirun* and the *gpu-launch.sh* MPI wrapper script.

Usage

```
Usage: mpirun [MPIRUN_OPTIONS] gpu-launch.sh [--gpu-list GPU_LIST] acecast.exe

    MPIRUN_OPTIONS: use "mpirun --help" for more information

    --gpu-list GPU_LIST (optional):

            This option can be used to specify which GPUs to use for running acecast. If
            running on multiple nodes then the list applies to all nodes. GPU_LIST should
            be a comma-separated list of non-negative integers or ranges (inclusive)
            corresponding to GPU device IDs. Examples:

                --gpu-list 0,1,3
                --gpu-list 0-2,4,6

            If this option is not provided then it is assumed that all detected GPUs are
            available for use and GPU_LIST will be determined automatically using the
            nvidia-smi utility.

            Note: GPU_LIST can also be set using the ACECAST_GPU_LIST environment␣
↪variable
```

Example 1

```
mpirun -np 3 ./gpu-launch.sh --gpu-list 0,1,3 ./acecast.exe
```

Namelist Support Table App

The namelist support table app can be used to search for available namelist options. Also be sure to Check out the *Namelist Advisor App* where you can upload your namelist and see if it is supported by AceCAST.

Not seeing your option(s)? *Let us know!*

Note: It may take a moment to load the application below.

Namelist Advisor App

Not seeing your option(s)? *Let us know!*

About

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

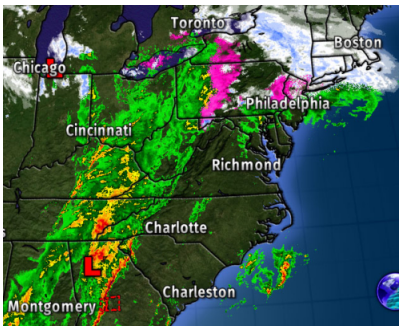
9. SPC convective outlooks
10. LSR (Local Storm Report) icons
11. METAR + NDBC surface observations
12. WPC Surface Analysis
13. GOES-R GLM lightning data with add-on subscription
14. Extensive forecast model data system
    - Defaults include all major NCEP models (GFS, NAM, HRRR, RAP, HIRESW)
    - 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

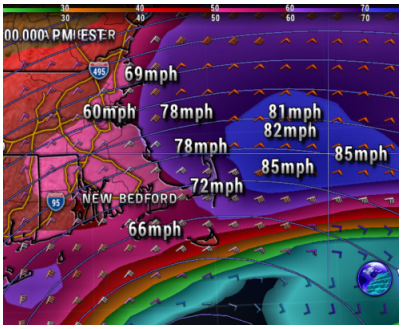
See the videos below for more details:

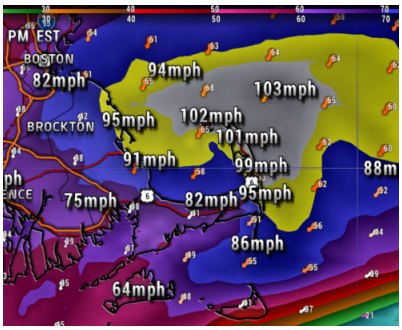
Gallery

Click each image for a larger view

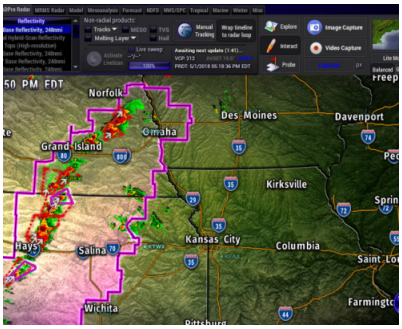
WSV3

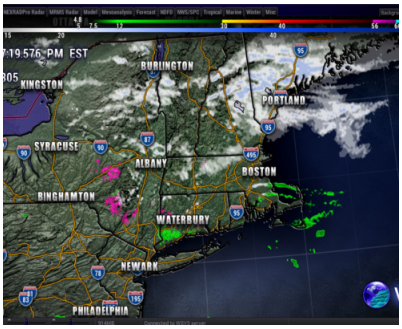


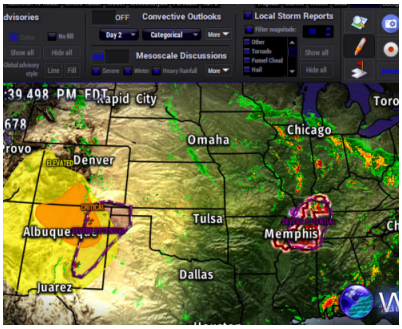

69mph
60mph
78mph
81mph
82mph
78mph
85mph
85mph
72mph
NEW BEDFORD
66mph


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

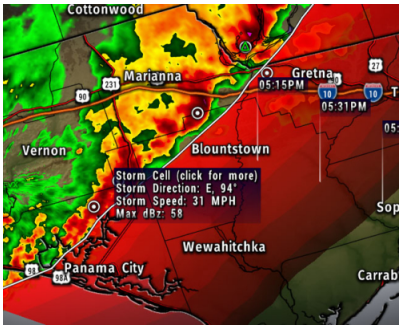

Storm ETA
6:03 PM EDT
40:45 until impact
Monroe
Arcadia
Jonesboro
Natchitoches
Vicksburg
Natchez
Yazoo City

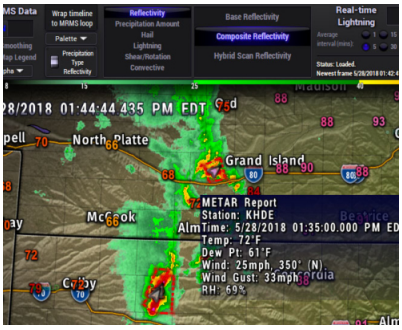

Norfolk
Omaha
Des Moines
Davenport
Grand Island
Kirksville
Kansas City
Columbia
Salina
Wichita


BURLINGTON
KINGSTON
PORTLAND
SYRACUSE
ALBANY
BOSTON
BINGHAMTON
WATERBURY
NEWARK
PHILADELPHIA


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


Memphis
Dallas
Jackson
Houston
New Orleans
Acuna
Monterrey

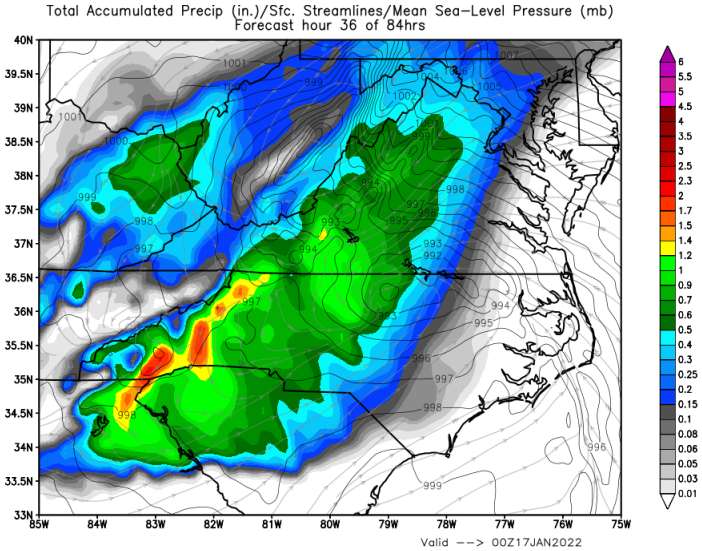

Cottonwood
Marianna
Gretna
05:15PM
05:31PM
Vernon
Blountstown
Storm Cell (click for more)
Storm Direction: E, 94°
Storm Speed: 31 MPH
Max dBz: 58
Wewahitchka
Panama City

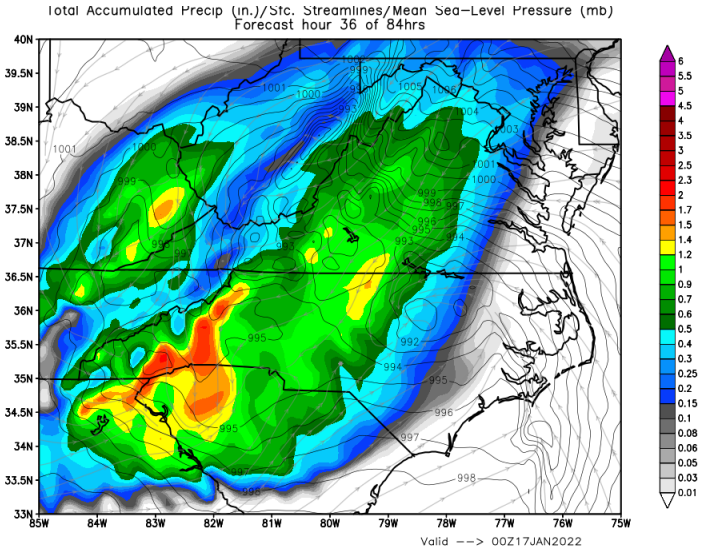

Reflectivity
Base Reflectivity
Composite Reflectivity
Real-time Lightning
North Platte
Grand Island
METAR Report
Station: KHDE
Temp: 72°F
Dew Pt: 61°F
Wind: 25mph, 350° (N)
Wind Gust: 33mph
RH: 69%


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

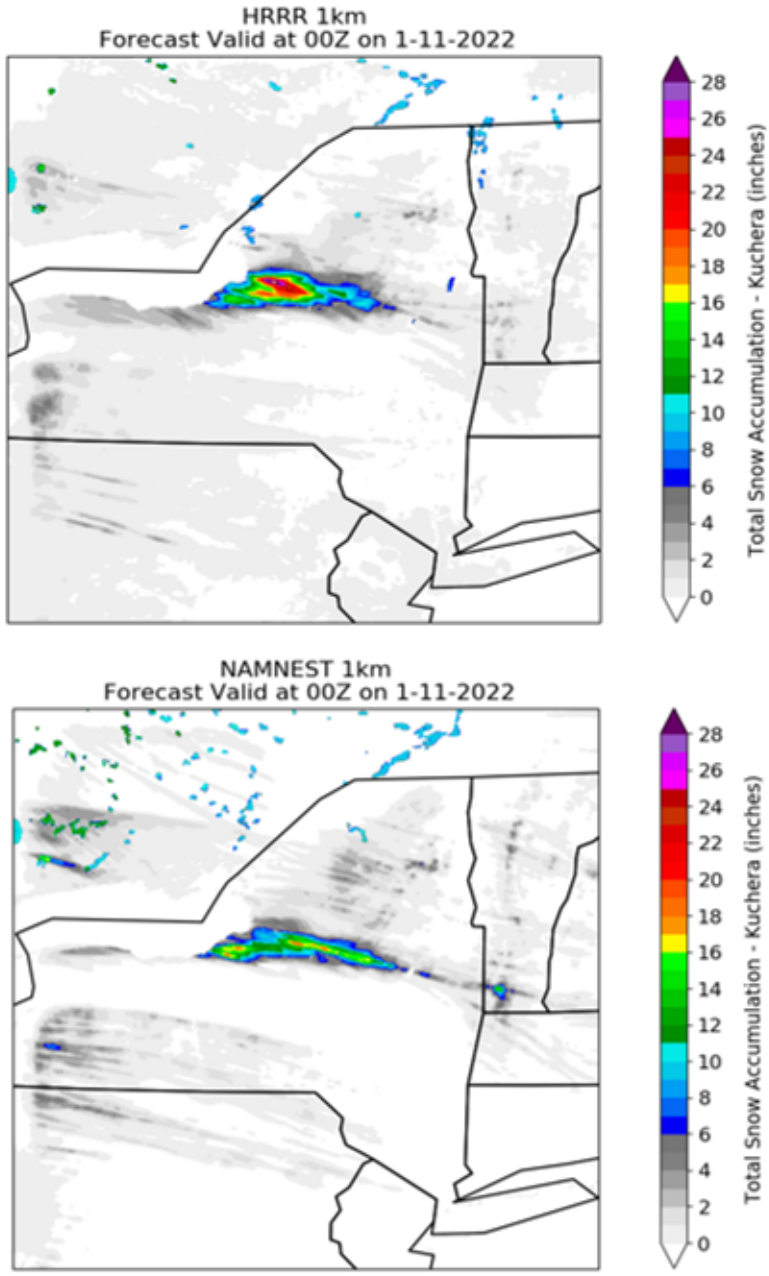
Winter Weather

NC/SC Snowstorm event January 2022

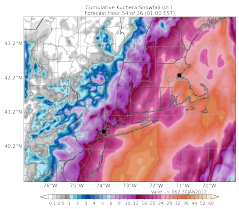


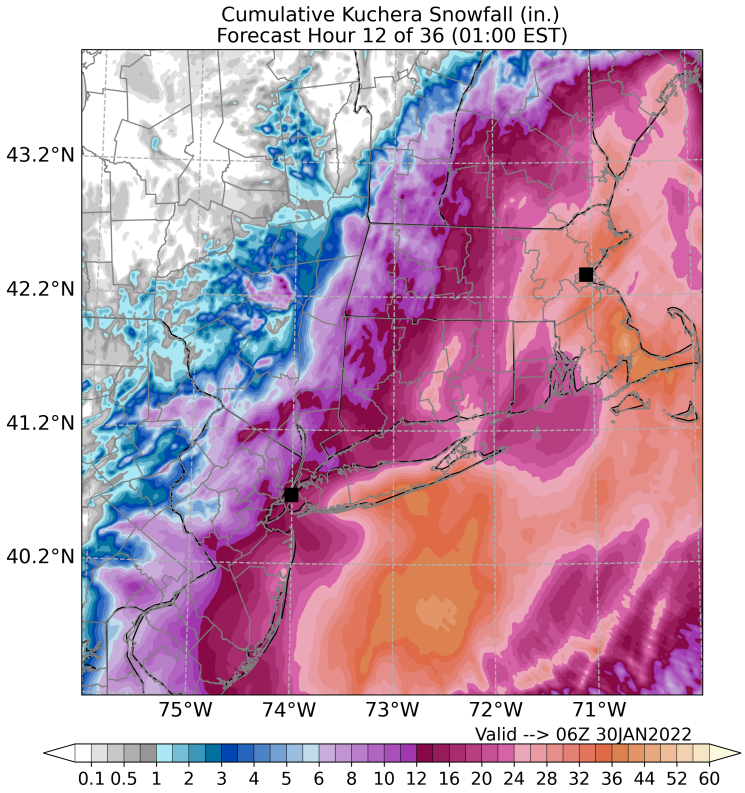


Upstate New York Lake Effect Snow



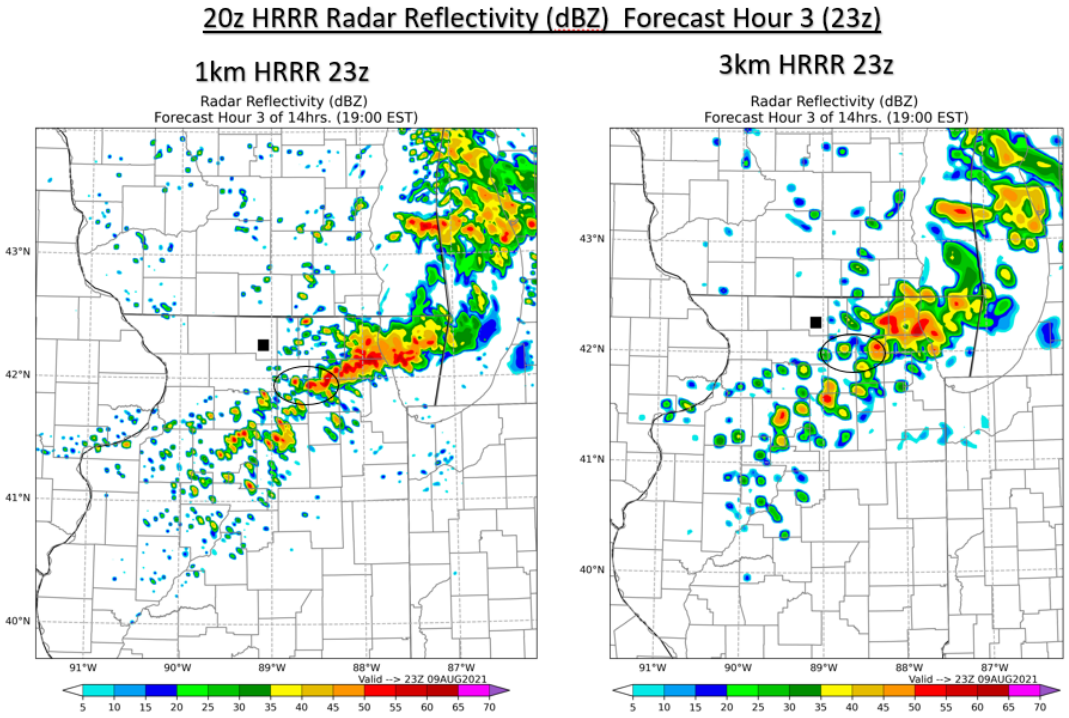
Northeast CONUS Snowstorm

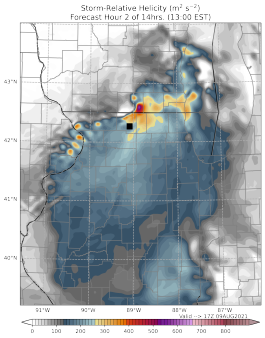


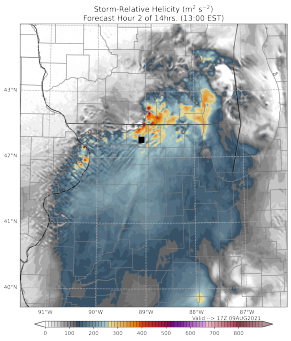


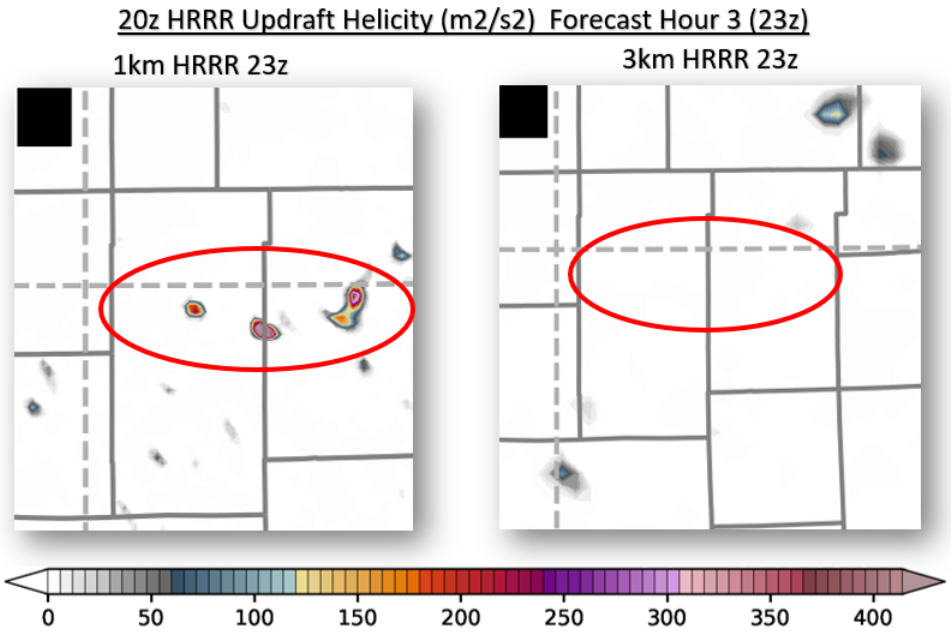
Severe Weather

Illinois Tornado Outbreak



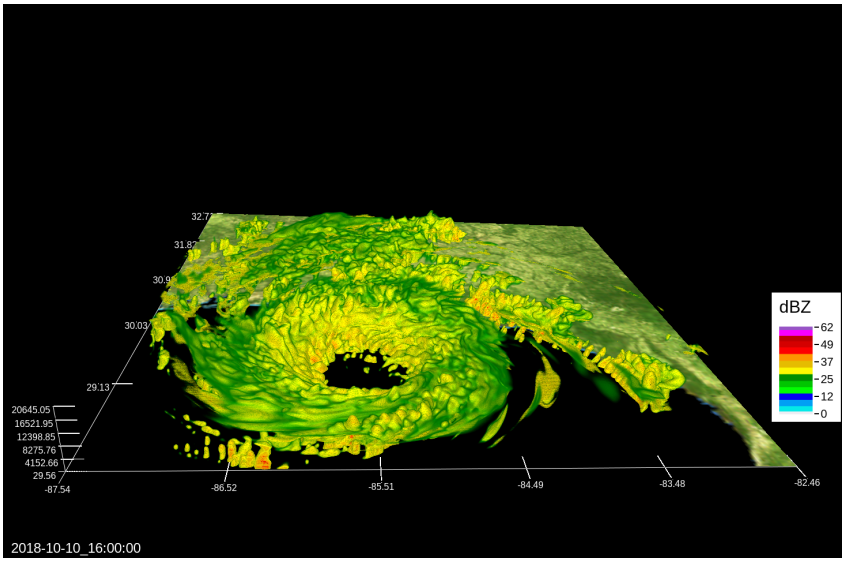


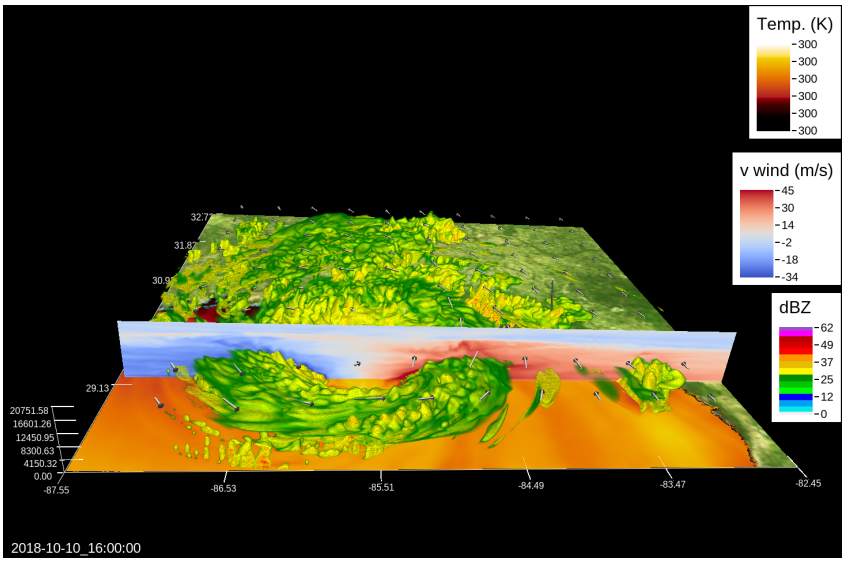




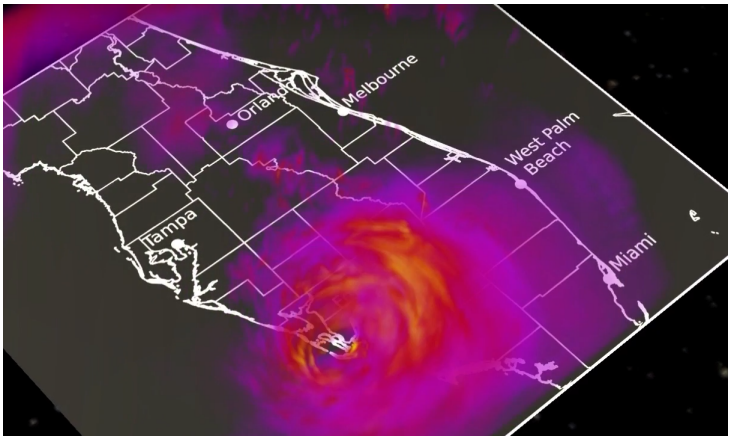
Tropical Weather

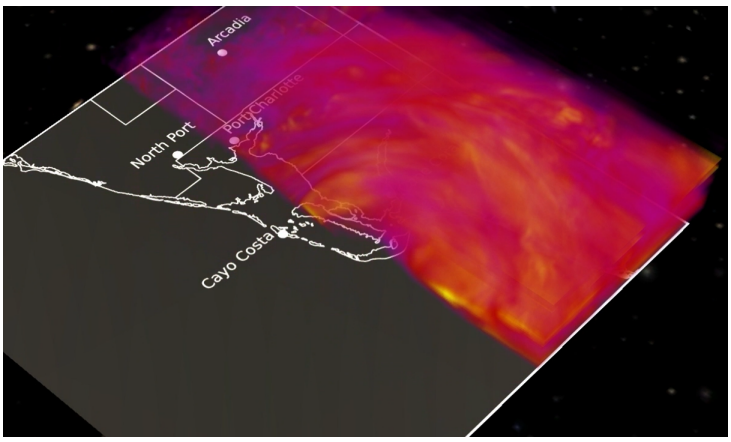
Hurricane Michael (2018)

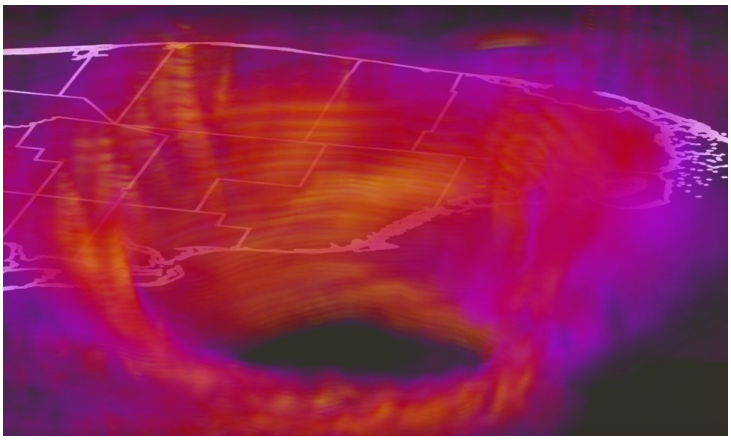


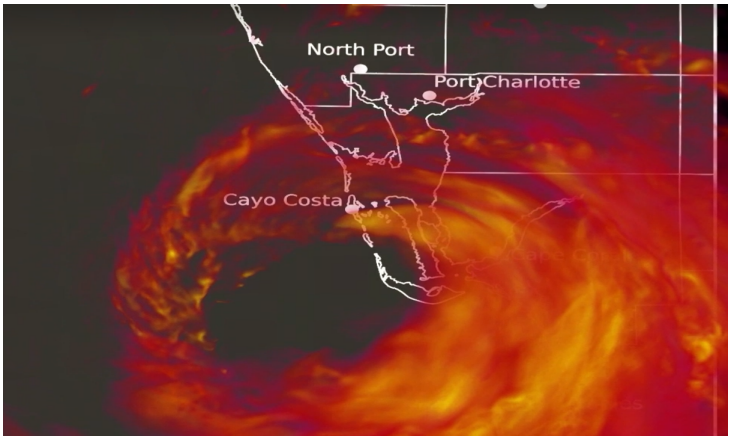


Hurricane Ian (2022)









Presentations

Below is a list of short presentations created by TempoQuest for additional information. Each link should download one PDF file to your computer:

- About TempoQuest
- About Numerical Weather Prediction (NWP)
- About AceCAST
- About AceCAST results
- About WRFOnDemand
- General
- AceCAST at WRF/MPAS 2023
- The Winds of Tomorrow - A Case Study

External Links

- Tempoquest homepage
- WRF homepage
- AceCAST OnDemand
- WSV3 data Visualizer
- Namelist.wps best practices
- Namelist.input best practices

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: Typically we have seen cost savings of 2-3x cheaper and runtime savings averaging x10 faster than CPU-equivalent simulations, however the exact value varies, please get in touch with *sales* for further details.

Runtime

1. Q: I downloaded AceCast, now what?
   - A: Check out the *Installation Guide* to get started!
2. Q: What are the system requirements?
   - A: Refer to *Platform Requirements*
3. Q: Do I need any other software to run AceCast?
   - A: There are some dependecies required. Please refer to *Installation Guide*
4. Q: Why is AceCAST running slower than WRF?
   - A: One reason could be, if you are using an older version of AceCAST you may see both a wrf.exe and an acecast.exe. Although both work, only acecast.exe is optimized for speed, wrf.exe was previously included only for user convenience.
5. Q: What kind of input is needed to run AceCAST?
   - A: AceCAST is WRF but on GPU's, so any compatible input to WRF should also work for AceCAST.

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, we would be happy to renew it for you!*
4. Q: WRF has many "submodels" which one's are supported by AceCAST?
    - A: Currently, TempoQuest only supports WRF-ARW, however we have plans to port WRF-DA and WRF-Chem to GPU's in the future. If you are interested in this please reach out to us and we can prioritize the porting.
5. Q: What hardware do I need to run AceCAST?
    - A: Since AceCAST is a GPU application, GPU's are needed. Theoretically any modern GPU would be compatible but we have found the best results and compatibility with "modern" HPC GPU's such as NVIDIA P100, V100, and A100 types.
6. Q: Can I run AceCAST in the cloud?
    - A: Yes! See our WRFOnDemand page to get started
7. Q: What namelist options are supported by AceCAST?
    - A: WRF (AceCAST) has a large array of options many of which are only used by a small percentage of users, thus we only support the major/most commonly used options. See the *Namelist Support Table App* to get a sense of which one's those are. If there is an option you are looking for that we do not yet support, we can port it for you.

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*

Verification and Validation

TempoQuest Verification Process

TempoQuest goes through an extensive review process to ensure their AceCAST product meets high quality standards. For every release of AceCAST TempoQuest runs AceCAST through 1,000s of small simulations to test for mainly different dynamics options and physics options. By comparing against 2 CPU baselines and checking the root mean squares and root mean squared errors for each default output variable, TempoQuest is able to verify that the results from AceCAST are the same as when running WRF on CPU's. Additionally, TempoQuest also checks that the results are the same (or better) across different versions of AceCAST. Through a series of regression tests, TempoQuest also checks AceCAST through 3 (main) different builds, different domain sizes and regions, different runtimes, different

GPU's (NVIDIA V100 and NVIDIA A100), support tests, host-compile/exec. preservation tests, bit-for-bit tests, and false-positive support tests. TempoQuest is proud to stand by their products' exceptional performance. Should you have any questions or concerns regarding the verification process and/or procedure, please feel free to reach out to us!

TempoQuest Validation Process

AceCAST by TempoQuest is built upon the Weather Research and Forecasting model (WRF). This means that results should be the same between the WRF and AceCAST. For any new features added to AceCAST TempoQuest checks the results of numerous variables (such as temperature, wind, radar, reflectivity, etc…) across 2 different CPU simulations and 1 GPU simulation. If the difference in values is within a specific range, the results are considered the same and thus depicts that AceCAST works as intended.

Disclaimer

TempoQuest is not responsible for bugs that appear in the base WRF model (CPU). This means that any problems with a simulation that appear in WRF simulation will likely also appear in AceCAST. However, TempoQuest submits any issues directly to the WRF developers to ensure that the problem is found, addressed, and fixed in a future release of the WRF model, and in extension, AceCAST.

Support

Under Construction

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

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.
```