

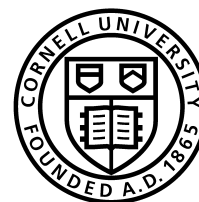


# Overview of I-WRF Container Architecture

Jared A. Lee & George McCabe (Presenters)  
NSF National Center for Atmospheric Research

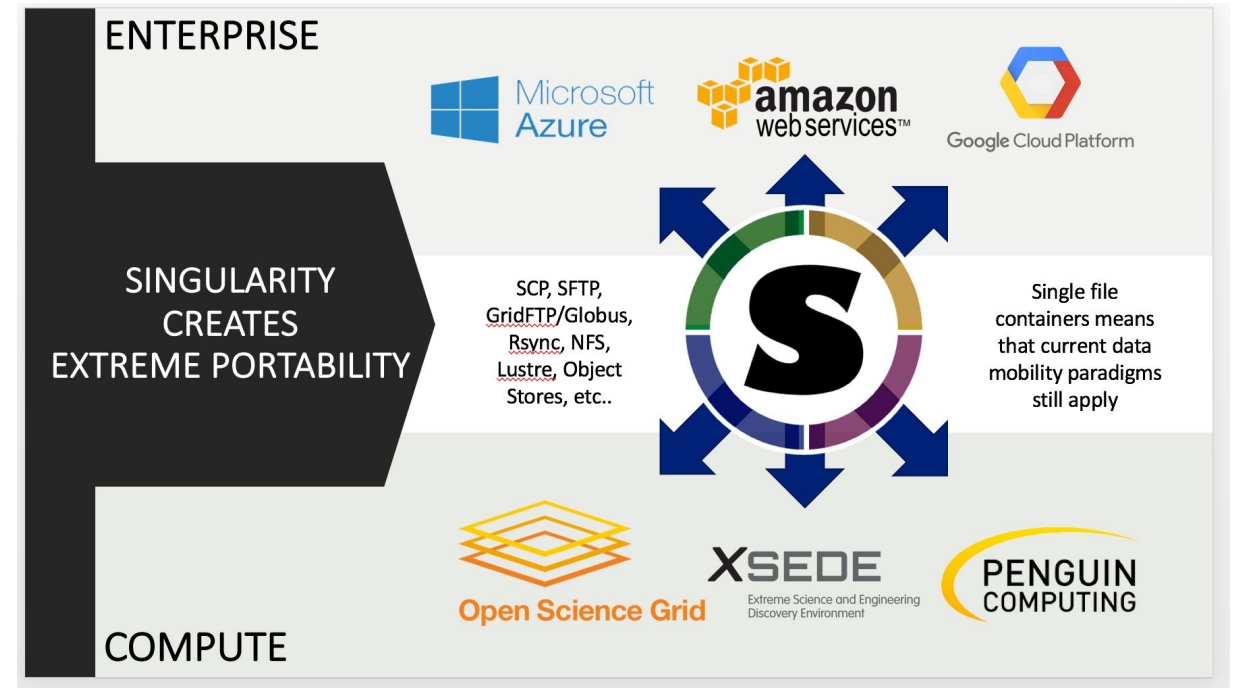
(On behalf of the entire project team from NSF NCAR & Cornell)

I-WRF Science Advisory Board Meeting  
20 August 2025

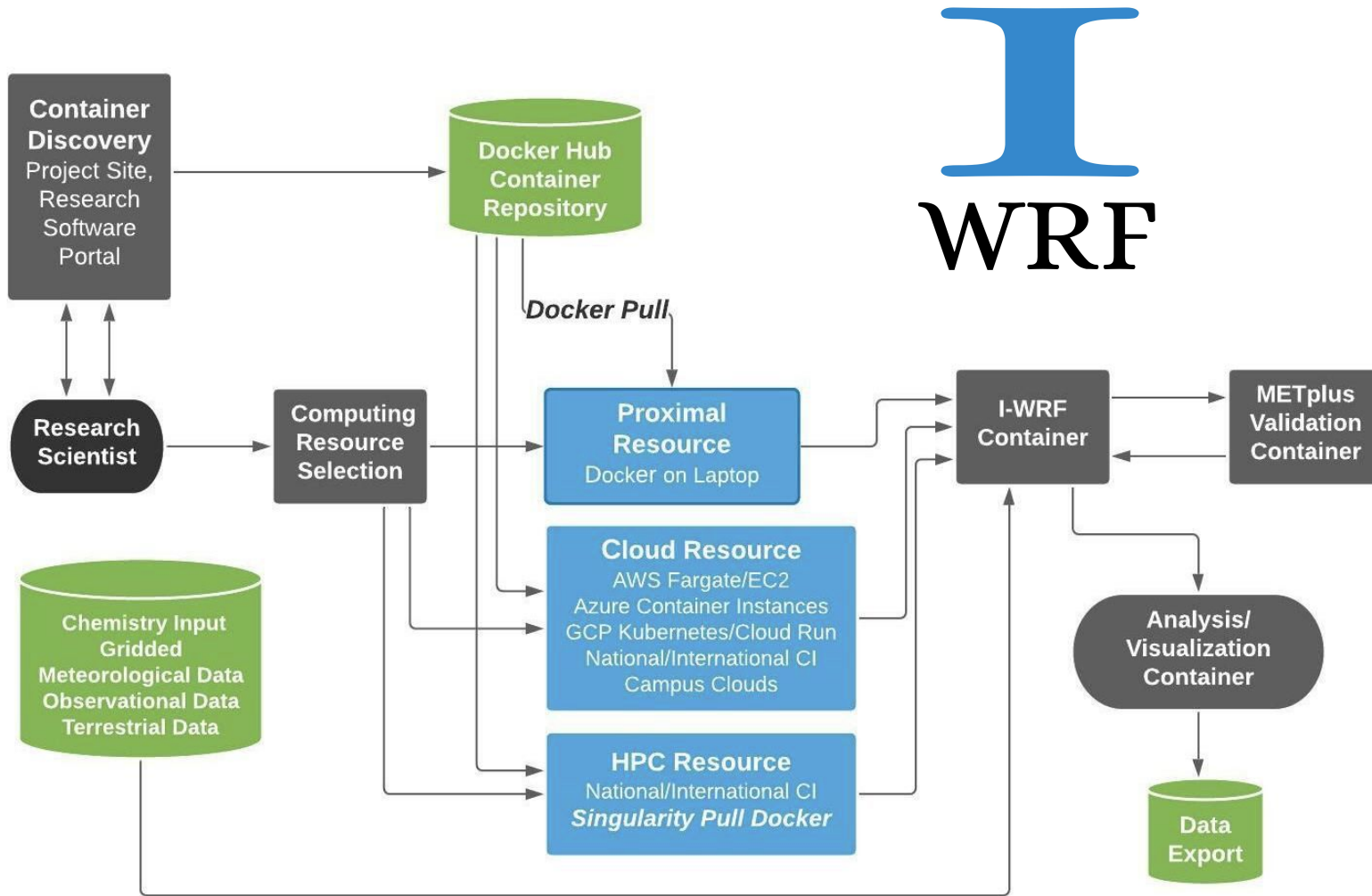


# Application Containerization

- An application can be put into a *software container* with all associated libraries and support
- The containerized application is smaller than a virtual machine image, and portable to several systems
- I-WRF puts the application, data, and configurations into a portable package

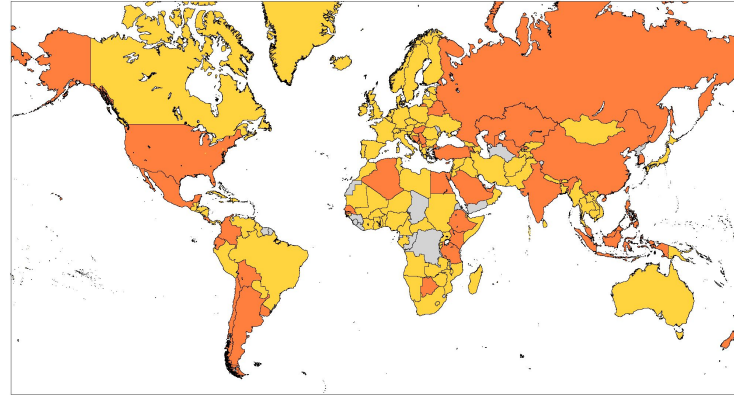
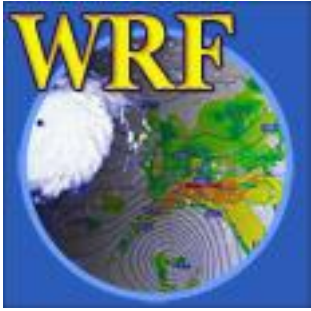


# I-WRF Conceptual Design



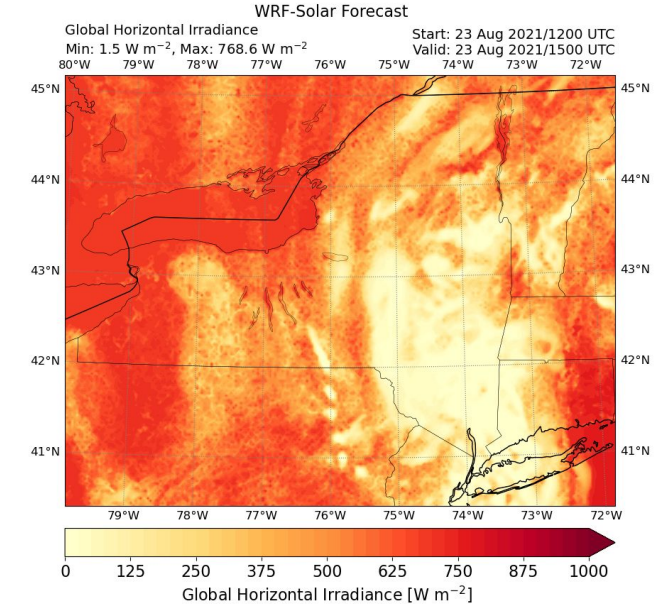
- The main change to this original concept diagram in our current implementation of I-WRF is that the visualization pieces are being done within the METplus validation container
- This way we only need to load in a Python environment in one container, not two

# Weather Research and Forecasting (WRF) model



Nations with registered WRF users  
Nations where WRF has been run  
operationally

Source: Fig. 2 in Powers et al. (2017, BAMS)



Source: Fig. 7a in Lee et al. (2024, Solar Energy)

- WRF® is a weather model with a broad range of applications
  - Weather prediction, regional climate modeling, real or idealized case studies
  - Simulation of events based on characteristics such as land use or cover
  - Chemistry/air quality, wildfire, renewable energy generation, hydrologic forecasting, crop growth modeling, aviation/turbulence, surface transportation, large-eddy simulation, and more
  - Validation and visualization tools for verifying and seeing results
- In development since 2000, with a user base of more than 30,000 worldwide
- Deployment across a wide range of HPC systems, so much as to be included in

# WRF Challenges

- Despite this, around *50% of users* attending tutorials at NSF NCAR report difficulty configuring the software for use on whichever computing platform they're using
- Compiling WRF software requires understanding multiple compiler frameworks, a set of required libraries to be built with the same compiler you select for WRF, and a wide range of WRF configuration options
- Need to know where to obtain data for initial conditions & lateral boundary conditions (ICs/LBCs), and observations for verification
- It usually requires some work to get verification and visualization tools configured to ingest WRF output
- These technical barriers mean that potential researchers and scholars run into hurdles before they can even get to the weather and climate stuff



Stanczyk, Jan Matejko, 1862.  
Wikimedia commons



<https://www.istockphoto.com/signature/photo/thats-it-im-done-qm936117884-256071691>



# Verifying Model Output with METplus

- Model Evaluation Tools (METplus) verification system
  - Community validation toolkit supported by NSF NCAR and largely developed by NSF NCAR
  - Developed through funding from the 557<sup>th</sup> Weather Wing of the U.S. Air Force, National Oceanic and Atmospheric Administration (NOAA), and NSF NCAR
  - Verification framework that spans a wide range of temporal (warn-on-forecast to climate) and spatial (storm to global) scales
  - Used operationally by NOAA, UK Met Office, Australian Bureau of Meteorology, and others
  - Large community of users & contributors
- METplus was already containerized
- I-WRF containerizes METplus configurations for doing some sample verification from the the various I-WRF use cases, and plots it

The logo for METplus, with 'MET' in large black letters and 'plus' in orange lowercase letters.

<https://dtcenter.org/community-code/metplus>

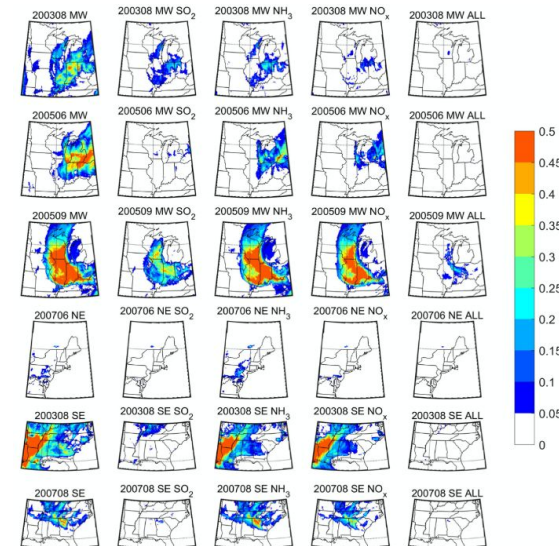
# I-WRF Goals

- **Application containers support simplicity, portability, and scalability**
  - Run on a wide range of systems without installation/configuration issues
  - Include data management and interoperability with validation and visualization tools
  - Allow for large-scale problems with multi-node processing
- **Another goal is to bring more researchers into Atmospheric Science**
  - I-WRF allows a user to try WRF without dealing with installing and compiling software
  - Model weather on your laptop, in the cloud, or on an HPC resource
    - Keep in mind, though, that your laptop doesn't have the computing horsepower of cloud or HPC resources, so the same simulation will take longer on a laptop



# I-WRF Science Use Cases — Running at scale to answer research questions

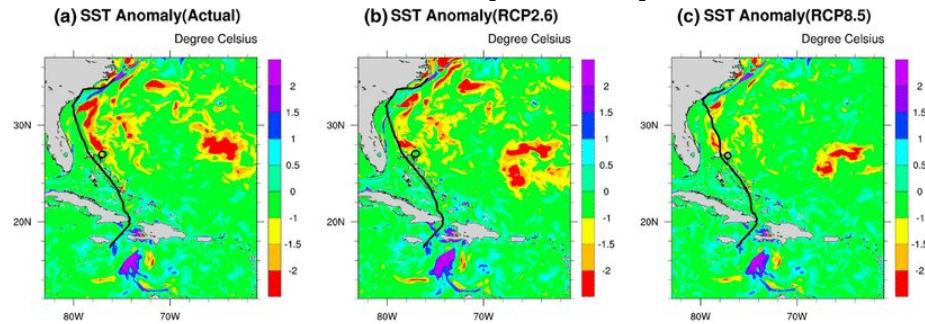
0. Hurricane Matthew (initial simple demo of containers)
1. Land Use/Land Cover (LULC) Change in the U.S. Northeast and Feedbacks to Extreme Weather Events and Societal Impacts
2. Climate Change Impacts on Wind and Solar Energy Resources in the U.S.
3. Air Quality in the Northeast U.S. Urban Corridor in a Changing Climate





# Supporting Broader Engagement in Atmospheric Science

- Users can run sample WRF simulations on a laptop or free cloud resource
- The first I-WRF sample simulation is an event used for the NSF NCAR Online WRF Tutorial: Hurricane Matthew (2016) event

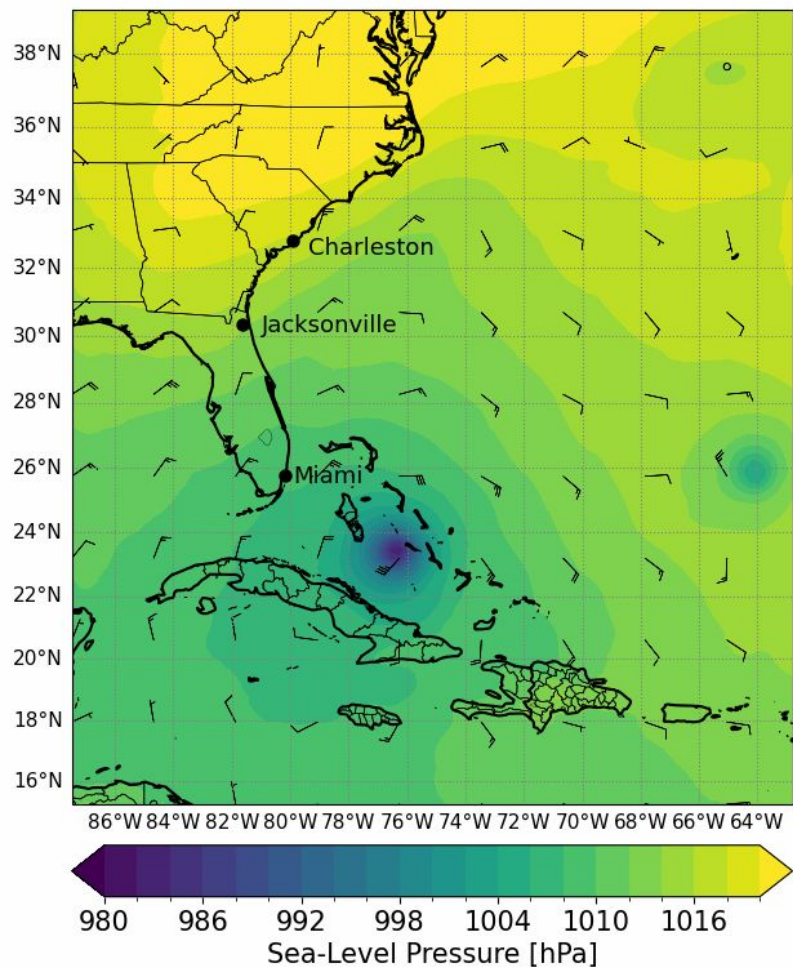


- Making the WRF software both easier to run and relevant to:
  - Increasing recruitment into Atmospheric Sciences
  - Building a **pipeline** of researchers into the discipline from a range of backgrounds

# I-WRF Hurricane Matthew Test Case Python Visualization

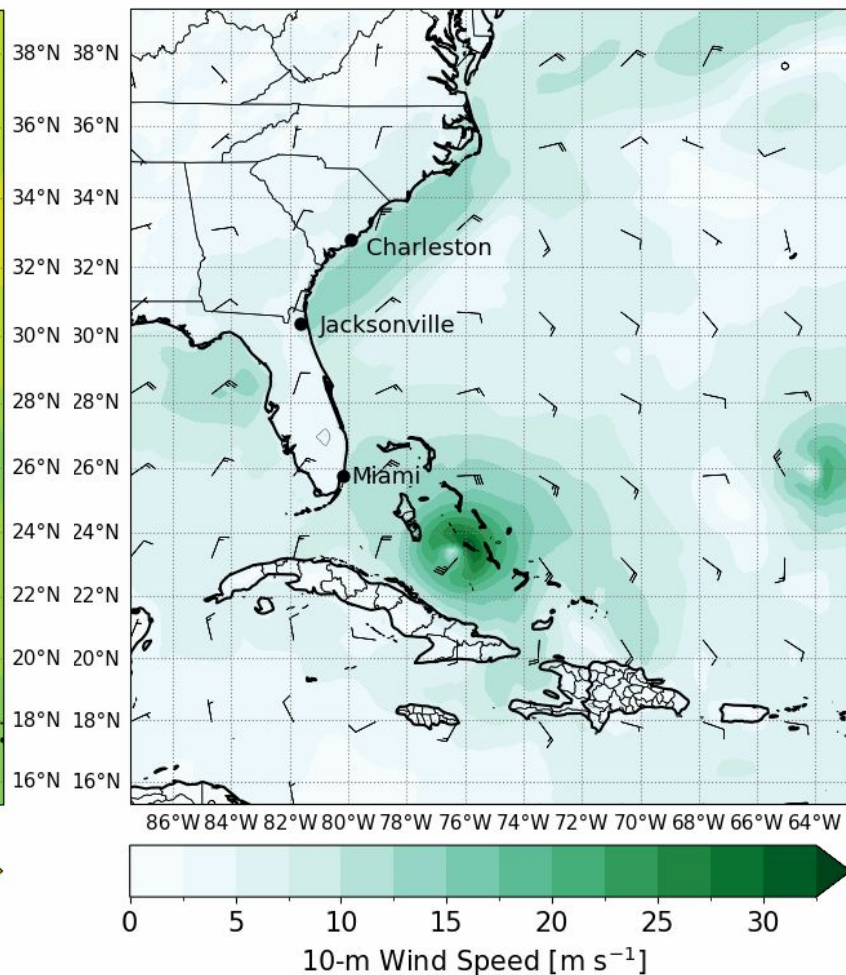
Hurricane Matthew Test Case

Sea-Level Pressure      Start: 06 Oct 2016/0000 UTC  
Min: 982.0 hPa, Max: 1025.0 hPa Valid: 06 Oct 2016/0300 UTC  
86°W 84°W 82°W 80°W 78°W 76°W 74°W 72°W 70°W 68°W 66°W 64°W



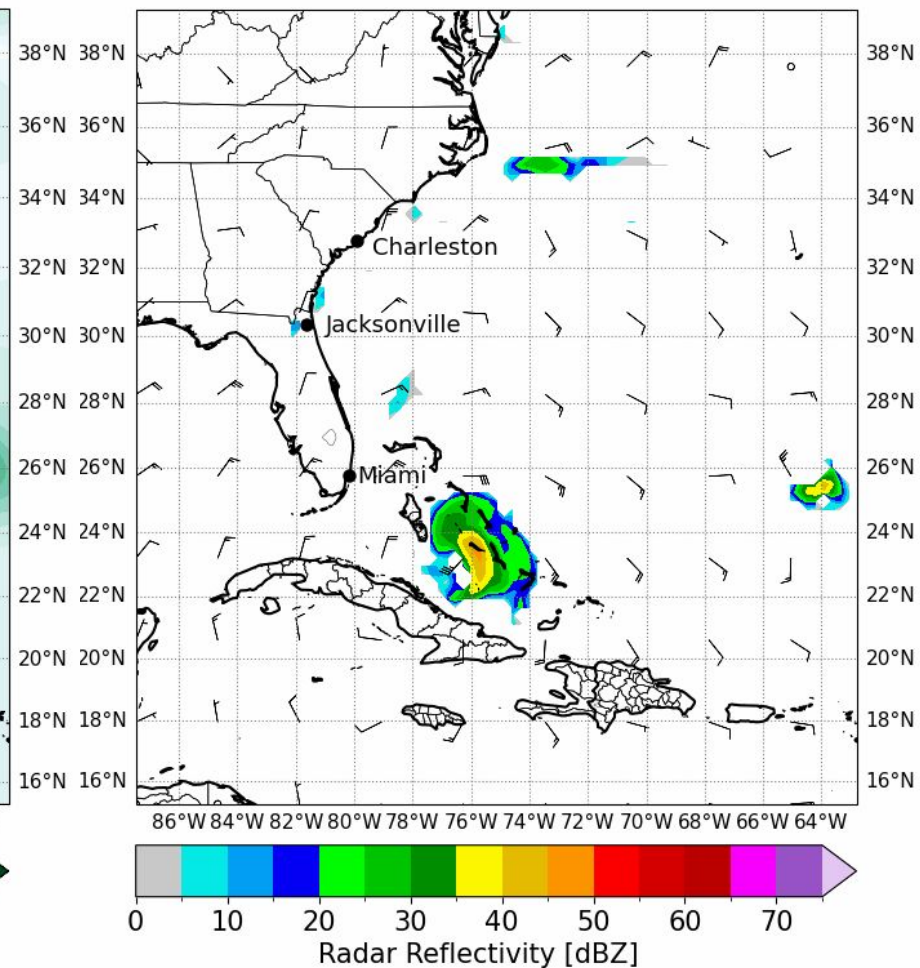
Hurricane Matthew Test Case

10-m Wind Speed      Start: 06 Oct 2016/0000 UTC  
Min: 0.0 m s<sup>-1</sup>, Max: 31.1 m s<sup>-1</sup> Valid: 06 Oct 2016/0300 UTC  
86°W 84°W 82°W 80°W 78°W 76°W 74°W 72°W 70°W 68°W 66°W 64°W



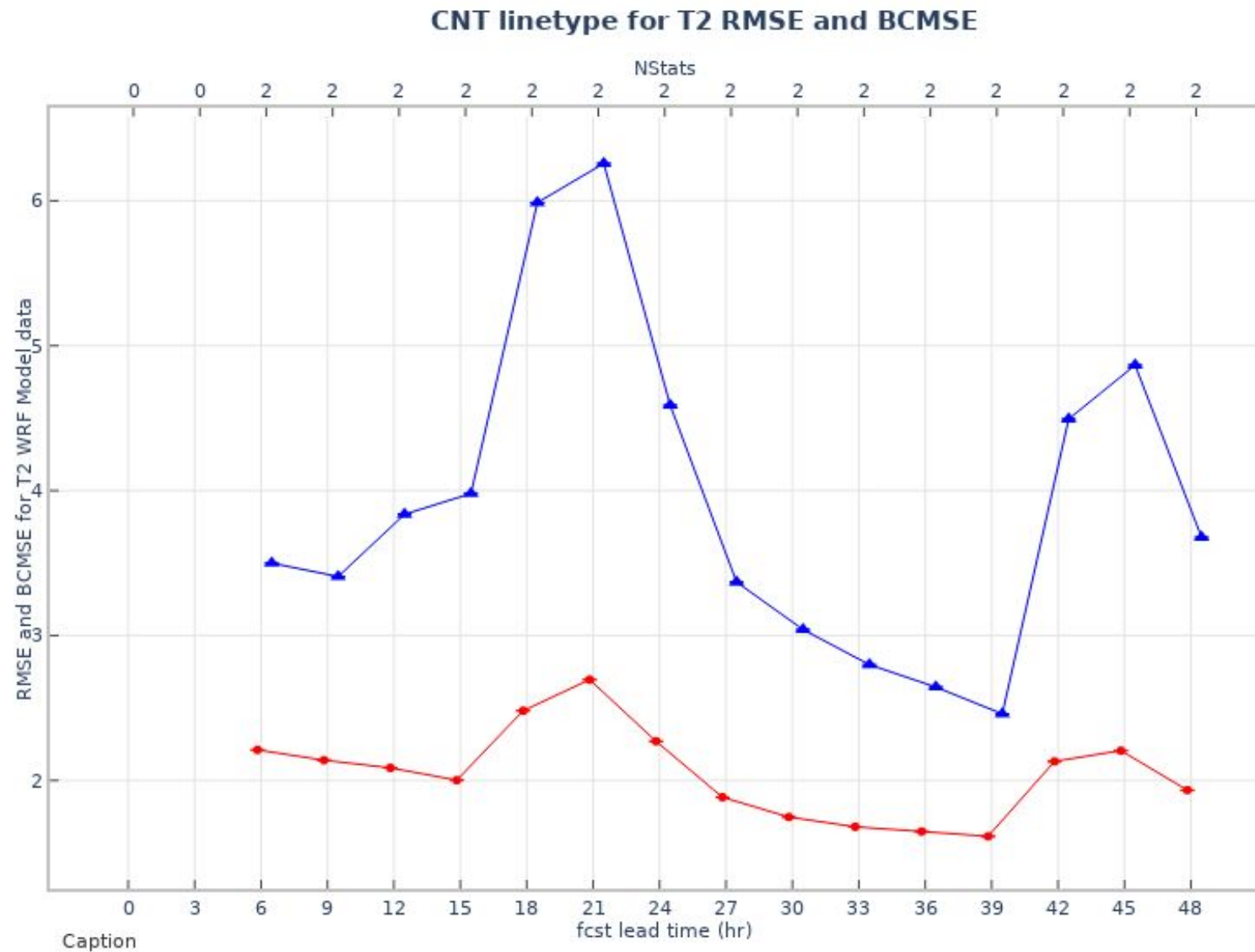
Hurricane Matthew Test Case

Radar Reflectivity; 10-m Barbs      Start: 06 Oct 2016/0000 UTC  
Min: -30.0 dBZ, Max: 46.8 dBZ Valid: 06 Oct 2016/0300 UTC  
86°W 84°W 82°W 80°W 78°W 76°W 74°W 72°W 70°W 68°W 66°W 64°W

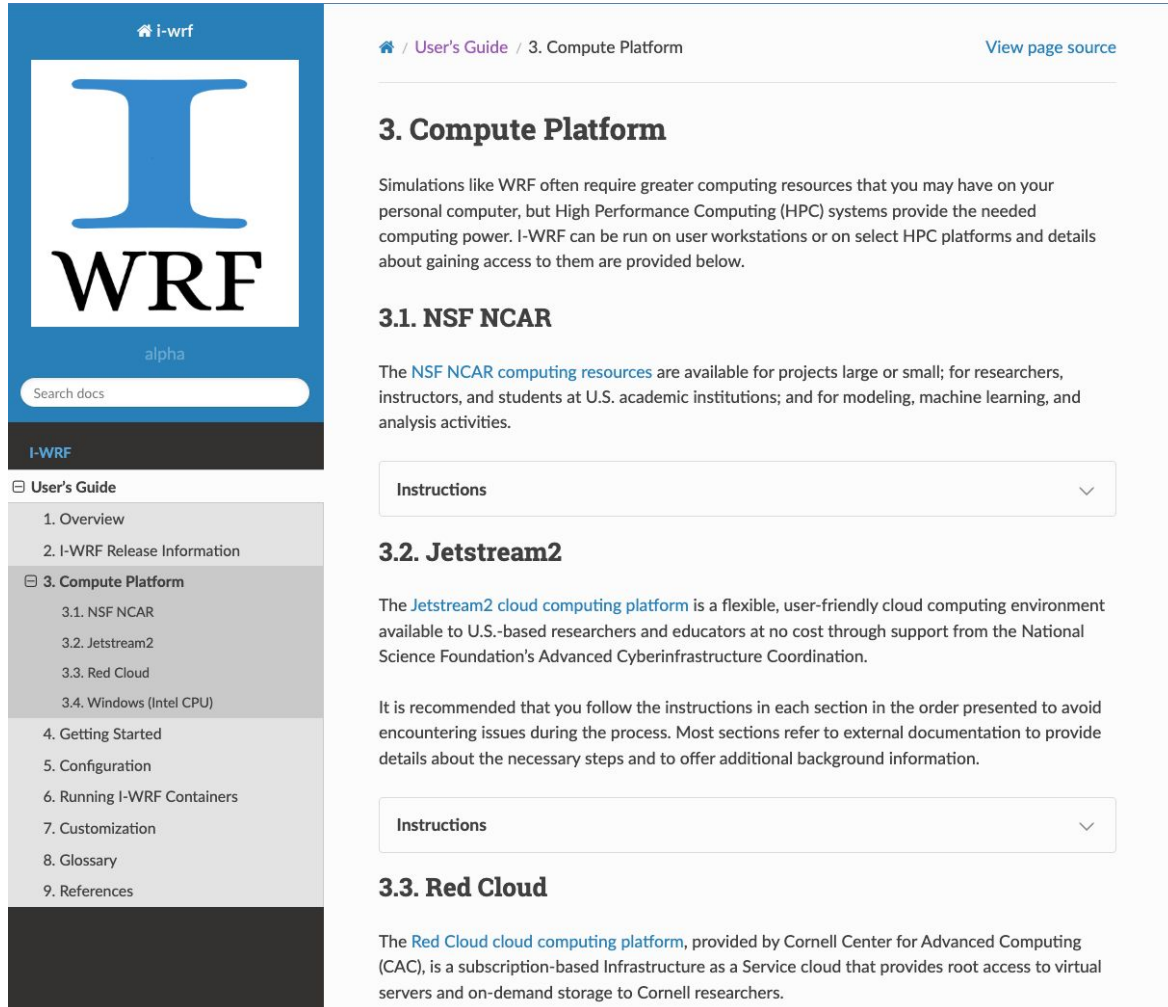




# I-WRF Hurricane Matthew METplus Visualization



# I-WRF ReadTheDocs Documentation



The screenshot shows the I-WRF documentation page for the '3. Compute Platform' section. The page has a blue header with the I-WRF logo and a search bar. The main content area is white with a blue sidebar on the left containing a table of contents. The '3. Compute Platform' section is highlighted in the sidebar. The main content area has a blue header with the I-WRF logo and a search bar. The main content area is white with a blue sidebar on the left containing a table of contents. The '3. Compute Platform' section is highlighted in the sidebar. The main content area has a blue header with the I-WRF logo and a search bar. The main content area is white with a blue sidebar on the left containing a table of contents. The '3. Compute Platform' section is highlighted in the sidebar.

## 3. Compute Platform

Simulations like WRF often require greater computing resources that you may have on your personal computer, but High Performance Computing (HPC) systems provide the needed computing power. I-WRF can be run on user workstations or on select HPC platforms and details about gaining access to them are provided below.

### 3.1. NSF NCAR

The [NSF NCAR computing resources](#) are available for projects large or small; for researchers, instructors, and students at U.S. academic institutions; and for modeling, machine learning, and analysis activities.

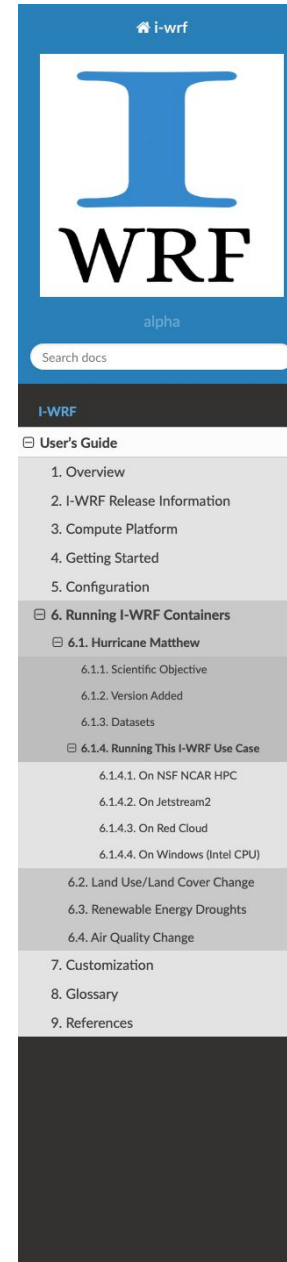
### 3.2. Jetstream2

The [Jetstream2 cloud computing platform](#) is a flexible, user-friendly cloud computing environment available to U.S.-based researchers and educators at no cost through support from the National Science Foundation's Advanced Cyberinfrastructure Coordination.

It is recommended that you follow the instructions in each section in the order presented to avoid encountering issues during the process. Most sections refer to external documentation to provide details about the necessary steps and to offer additional background information.

### 3.3. Red Cloud

The [Red Cloud cloud computing platform](#), provided by Cornell Center for Advanced Computing (CAC), is a subscription-based Infrastructure as a Service cloud that provides root access to virtual servers and on-demand storage to Cornell researchers.



The screenshot shows the I-WRF documentation page for the '6.1.4. Running This I-WRF Use Case' section. The page has a blue header with the I-WRF logo and a search bar. The main content area is white with a blue sidebar on the left containing a table of contents. The '6.1.4. Running This I-WRF Use Case' section is highlighted in the sidebar. The main content area has a blue header with the I-WRF logo and a search bar. The main content area is white with a blue sidebar on the left containing a table of contents. The '6.1.4. Running This I-WRF Use Case' section is highlighted in the sidebar.

## 6.1.4. Running This I-WRF Use Case

With your instance created and running and you logged in to it, you can now install the necessary software and download the data to run the simulation.

Instructions are provided below for running the Hurricane Matthew use case for each [Compute Platform](#) on which it has been tested.

### 6.1.4.1. On NSF NCAR HPC

Follow the compute platform instructions for [NSF NCAR](#) to secure access to and log in to NSF NCAR HPC.

These instructions are currently limited to running the METplus verification software and assume that WRF output is already available in a local directory.

## 6.1.4. Running This I-WRF Use Case

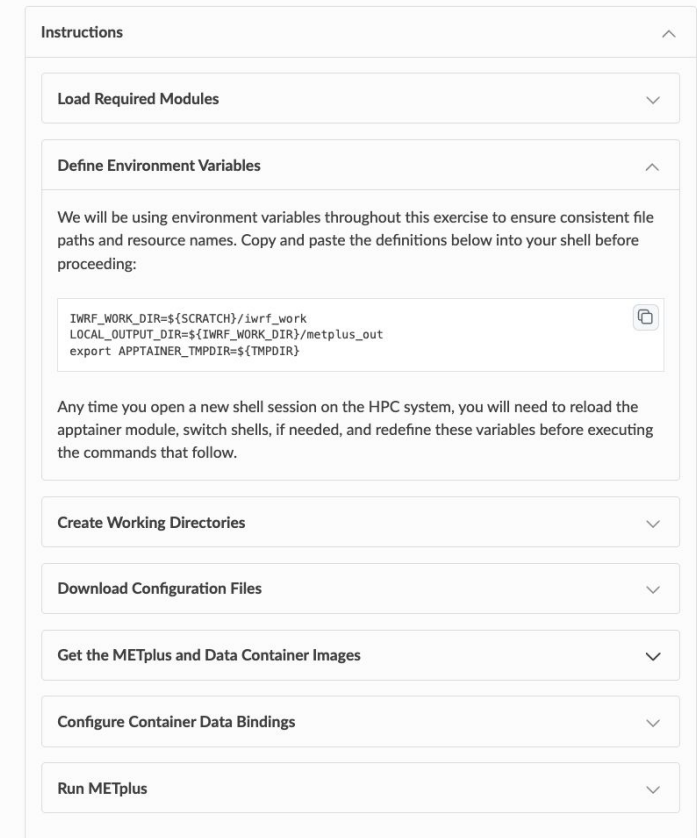
With your instance created and running and you logged in to it, you can now install the necessary software and download the data to run the simulation.

Instructions are provided below for running the Hurricane Matthew use case for each [Compute Platform](#) on which it has been tested.

### 6.1.4.1. On NSF NCAR HPC

Follow the compute platform instructions for [NSF NCAR](#) to secure access to and log in to NSF NCAR HPC.

These instructions are currently limited to running the METplus verification software and assume that WRF output is already available in a local directory.



The screenshot shows the I-WRF documentation page for the '6.1.4.1. On NSF NCAR HPC' section. The page has a blue header with the I-WRF logo and a search bar. The main content area is white with a blue sidebar on the left containing a table of contents. The '6.1.4.1. On NSF NCAR HPC' section is highlighted in the sidebar. The main content area has a blue header with the I-WRF logo and a search bar. The main content area is white with a blue sidebar on the left containing a table of contents. The '6.1.4.1. On NSF NCAR HPC' section is highlighted in the sidebar.

## 6.1.4.1. On NSF NCAR HPC

Follow the compute platform instructions for [NSF NCAR](#) to secure access to and log in to NSF NCAR HPC.

These instructions are currently limited to running the METplus verification software and assume that WRF output is already available in a local directory.

### Instructions

- Load Required Modules
- Define Environment Variables
- Create Working Directories
- Download Configuration Files
- Get the METplus and Data Container Images
- Configure Container Data Bindings
- Run METplus

### 6.1.4.2. On Jetstream2

- User guide:  
[https://i-wrf.readthedocs.io/en/latest/Users\\_Guide/index.html](https://i-wrf.readthedocs.io/en/latest/Users_Guide/index.html)



# Running the Hurricane Matthew Use Case

```

0L1SS (priority=1, resolution='default', path='/home/wrfuser/terrestrial_data/WPS_GEOG/orogwd3_10m/ol1ss/')
0L2SS (priority=1, resolution='default', path='/home/wrfuser/terrestrial_data/WPS_GEOG/orogwd3_10m/ol2ss/')
0L3SS (priority=1, resolution='default', path='/home/wrfuser/terrestrial_data/WPS_GEOG/orogwd3_10m/ol3ss/')
0L4SS (priority=1, resolution='default', path='/home/wrfuser/terrestrial_data/WPS_GEOG/orogwd3_10m/ol4ss/')
BATHYMETRY (priority=1, resolution='default', path='/home/wrfuser/terrestrial_data/WPS_GEOG/topobath_30s/')
MH_URB2D (priority=1, resolution='', path='')
ZD_URB2D (priority=1, resolution='', path='')
Z0_URB2D (priority=1, resolution='', path='')
BUILD_AREA_FRACTION (priority=1, resolution='', path='')
LF_URB2D_S (priority=1, resolution='', path='')
AHE (priority=1, resolution='', path='')
AHE (priority=2, resolution='', path='')

```

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!  
! Successful completion of geogrid.          !  
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

```
MPI startup(): I_MPI_OFI_LIBRARY variable has been removed from the product, its value is ignored
```

Processing domain 1 of 1

Processing 2016-10-06\_00

FILE

Processing 2016-10-06\_06

FILE

Processing 2016-10-06\_12  
FILE

FILE  
Processing 2016 10 06 10

Processing 2016-10-06\_18  
ETLE

Processing 2016-10-07 00

FILE

Processing 2016-10-07 06

FILE

Processing 2016-10-07\_12

FILE

Processing 2016-10-07\_18

FILE

Processing 2016-10-08\_00

FILE \_\_\_\_\_

```
.....
| Successful completion
```

```

: Successful completion
|||||

```

```
.....
MPT_startup(): T MPT OFT
```

$$f(x) = \begin{cases} 1 & \text{if } x \in \mathbb{Q} \\ 0 & \text{if } x \notin \mathbb{Q} \end{cases}$$

```
starting wrf task
```

```
starting wrf task
```

```
starting wrf task
```

```
starting wrf task
```

```
starting wrf task
```

```
starting wrf task
starting wrf task
```

```
starting wrf task
starting wrf task
```

```
starting wtf task
MPT_startup(): T MPT OKT
```

```
MF1_startcup(). 1_MF1_of 1_
```

```
starting wrf task
```

Starting in 1998, each

```
jaredlee@dec2343:/glade/derecho/scratch/jaredlee/iwrf_work> apptainer exec ${WORKING_DIR}/iwrf-metplus.sif /metplus/METplus/
sh/run_metplus.py /config/PointStat_matthew.conf
Starting METplus v6.0.0-rc1
Parsing config file: /metplus/METplus/metplus/parm/metplus_config/defaults.conf
Parsing config file: /config/PointStat_matthew.conf
Logging to /data/output/logs/metplus.log.20250818211015
08/18 21:10:15.042Z metplus.1033ee1d INFO: Running METplus v6.0.0-rc1 as user jaredlee(14991) with command: /metplus/METplus/
ush/run_metplus.py /config/PointStat_matthew.conf
08/18 21:10:15.044Z metplus.1033ee1d INFO: Log file: /data/output/logs/metplus.log.20250818211015
08/18 21:10:15.044Z metplus.1033ee1d INFO: METplus Base: /metplus/METplus
08/18 21:10:15.044Z metplus.1033ee1d INFO: Final Conf: /data/output/metplus_final.conf.20250818211015
08/18 21:10:15.044Z metplus.1033ee1d INFO: Config Input: /metplus/METplus/metplus/parm/metplus_config/defaults.conf
08/18 21:10:15.044Z metplus.1033ee1d INFO: Config Input: /config/PointStat_matthew.conf
08/18 21:10:15.066Z metplus.1033ee1d INFO: Running wrapper: MADIS2NC(metar)
08/18 21:10:15.067Z metplus.1033ee1d INFO: *****
08/18 21:10:15.068Z metplus.1033ee1d INFO: * Running METplus MADIS2NCWrapper(metar)
08/18 21:10:15.068Z metplus.1033ee1d INFO: * at init time: 2016-10-06 00:00
08/18 21:10:15.068Z metplus.1033ee1d INFO: * *****
08/18 21:10:15.069Z metplus.1033ee1d INFO: Processing forecast lead 0 hours
08/18 21:10:15.071Z metplus.1033ee1d INFO: COMMAND: /usr/local/bin/madis2nc /data/input/obs/metar/20161006_0000 /data/output/
madis2nc/metar/met_20161006_0000.nc -type metar -config /metplus/METplus/metplus/parm/met_config/Madis2NcConfig_wrapped -v 2
08/18 21:10:18.686Z metplus.1033ee1d INFO: Finished running /usr/local/bin/madis2nc - took 0:00:03.608394
08/18 21:10:18.687Z metplus.1033ee1d INFO: Processing forecast lead 1 hour
08/18 21:10:18.690Z metplus.1033ee1d INFO: COMMAND: /usr/local/bin/madis2nc /data/input/obs/metar/20161006_0100 /data/output/
madis2nc/metar/met_20161006_0100.nc -type metar -config /metplus/METplus/metplus/parm/met_config/Madis2NcConfig_wrapped -v 2
08/18 21:10:21.296Z metplus.1033ee1d INFO: Finished running /usr/local/bin/madis2nc - took 0:00:02.605761
08/18 21:10:21.297Z metplus.1033ee1d INFO: Processing forecast lead 2 hours
08/18 21:10:21.300Z metplus.1033ee1d INFO: COMMAND: /usr/local/bin/madis2nc /data/input/obs/metar/20161006_0200 /data/output/
madis2nc/metar/met_20161006_0200.nc -type metar -config /metplus/METplus/metplus/parm/met_config/Madis2NcConfig_wrapped -v 2
08/18 21:10:23.906Z metplus.1033ee1d INFO: Finished running /usr/local/bin/madis2nc - took 0:00:02.605675
08/18 21:10:23.907Z metplus.1033ee1d INFO: Processing forecast lead 3 hours
08/18 21:10:23.910Z metplus.1033ee1d INFO: COMMAND: /usr/local/bin/madis2nc /data/input/obs/metar/20161006_0300 /data/output/
madis2nc/metar/met_20161006_0300.nc -type metar -config /metplus/METplus/metplus/parm/met_config/Madis2NcConfig_wrapped -v 2
08/18 21:10:26.517Z metplus.1033ee1d INFO: Finished running /usr/local/bin/madis2nc - took 0:00:02.605745
08/18 21:10:26.517Z metplus.1033ee1d INFO: Processing forecast lead 4 hours
08/18 21:10:26.520Z metplus.1033ee1d INFO: COMMAND: /usr/local/bin/madis2nc /data/input/obs/metar/20161006_0400 /data/output/
madis2nc/metar/met_20161006_0400.nc -type metar -config /metplus/METplus/metplus/parm/met_config/Madis2NcConfig_wrapped -v 2
08/18 21:10:29.327Z metplus.1033ee1d INFO: Finished running /usr/local/bin/madis2nc - took 0:00:02.806033
08/18 21:10:29.327Z metplus.1033ee1d INFO: Processing forecast lead 5 hours
08/18 21:10:29.330Z metplus.1033ee1d INFO: COMMAND: /usr/local/bin/madis2nc /data/input/obs/metar/20161006_0500 /data/output/
madis2nc/metar/met_20161006_0500.nc -type metar -config /metplus/METplus/metplus/parm/met_config/Madis2NcConfig_wrapped -v 2
08/18 21:10:31.936Z metplus.1033ee1d INFO: Finished running /usr/local/bin/madis2nc - took 0:00:02.605705
08/18 21:10:31.937Z metplus.1033ee1d INFO: Processing forecast lead 6 hours
08/18 21:10:31.941Z metplus.1033ee1d INFO: COMMAND: /usr/local/bin/madis2nc /data/input/obs/metar/20161006_0600 /data/output/
madis2nc/metar/met_20161006_0600.nc -type metar -config /metplus/METplus/metplus/parm/met_config/Madis2NcConfig_wrapped -v 2
08/18 21:10:34.748Z metplus.1033ee1d INFO: Finished running /usr/local/bin/madis2nc - took 0:00:02.805935
08/18 21:10:34.748Z metplus.1033ee1d INFO: Processing forecast lead 7 hours
08/18 21:10:34.750Z metplus.1033ee1d INFO: COMMAND: /usr/local/bin/madis2nc /data/input/obs/metar/20161006_0700 /data/output/
madis2nc/metar/met_20161006_0700.nc -type metar -config /metplus/METplus/metplus/parm/met_config/Madis2NcConfig_wrapped -v 2
08/18 21:10:37.557Z metplus.1033ee1d INFO: Finished running /usr/local/bin/madis2nc - took 0:00:02.805948
08/18 21:10:37.557Z metplus.1033ee1d INFO: Processing forecast lead 8 hours
08/18 21:10:37.559Z metplus.1033ee1d INFO: COMMAND: /usr/local/bin/madis2nc /data/input/obs/metar/20161006_0800 /data/output/
madis2nc/metar/met_20161006_0800.nc -type metar -config /metplus/METplus/metplus/parm/met_config/Madis2NcConfig_wrapped -v 2
08/18 21:10:40.366Z metplus.1033ee1d INFO: Finished running /usr/local/bin/madis2nc - took 0:00:02.805913
08/18 21:10:40.366Z metplus.1033ee1d INFO: Processing forecast lead 9 hours
```



# I-WRF Tutorial at MS-CC Workshop at Alabama A&M University

- Ben Trumbore (Cornell) and Jared Lee (NSF NCAR) gave an I-WRF Tutorial at an MS-CC Workshop in Oct 2024
- Mix of undergrads, grad students, & postdocs
- One student worked through the instructions himself that night at his hotel, and followed it without any issues



About Us ▾

Get Involved ▾

Community Resources ▾

News

Contact



Join

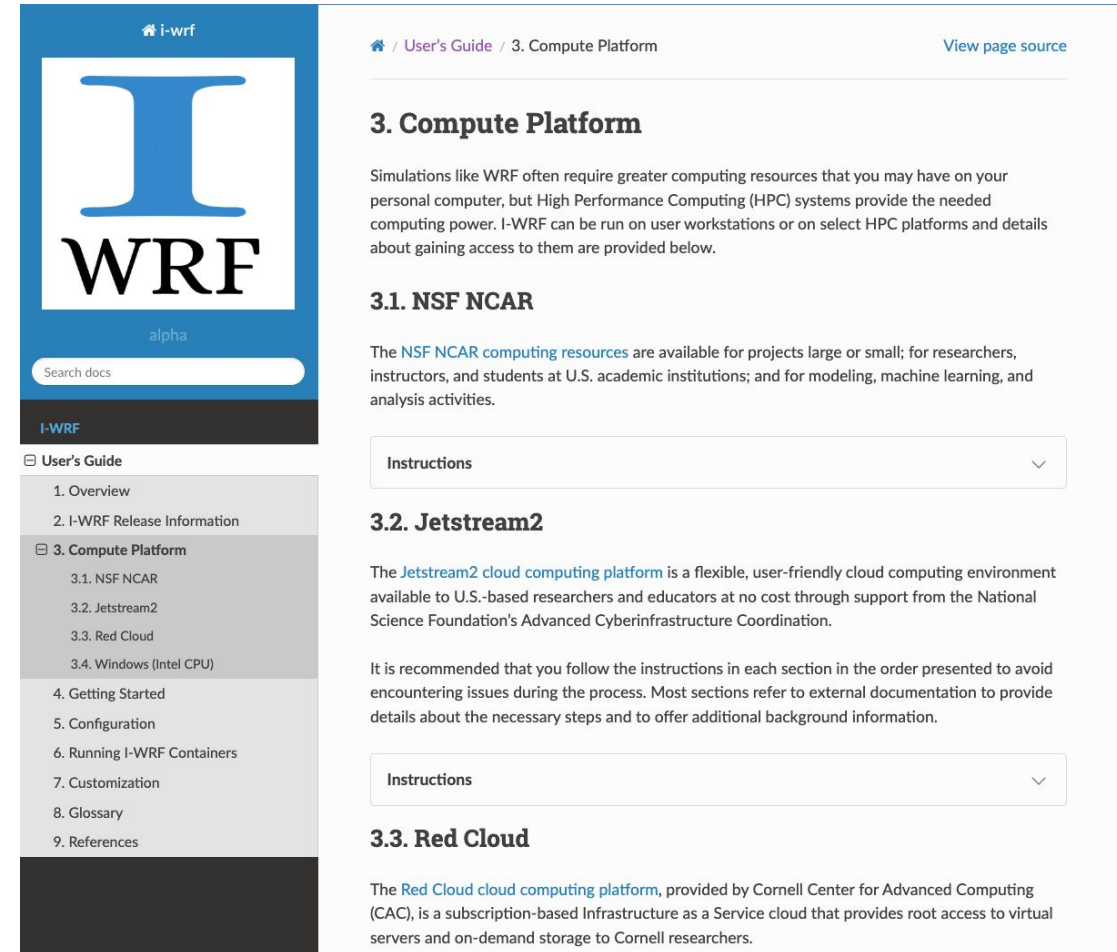
## MS-CC Workshop: Campus Technology, Cybersecurity, & Research Computing Support

Alabama A&M University  
Oct 29–30, 2024



# I-WRF Details

- Run it yourself on Derecho, Jetstream2, RedCloud2, or Windows:  
[https://i-wrf.readthedocs.io/en/latest/Users\\_Guide/use-cases/matthew.html](https://i-wrf.readthedocs.io/en/latest/Users_Guide/use-cases/matthew.html)
- Overview website: <https://i-wrf.org>
- User guide:  
[https://i-wrf.readthedocs.io/en/latest/Users\\_Guide/index.html](https://i-wrf.readthedocs.io/en/latest/Users_Guide/index.html)
- Github site:  
<https://github.com/NCAR/i-wrf>
- Help through [help@cac.cornell.edu](mailto:help@cac.cornell.edu)



The screenshot displays the I-WRF documentation website. The header features the I-WRF logo and a search bar. The left sidebar contains a navigation menu with the following items: 1. Overview, 2. I-WRF Release Information, 3. Compute Platform (selected), 4. Getting Started, 5. Configuration, 6. Running I-WRF Containers, 7. Customization, 8. Glossary, and 9. References. The main content area shows the '3. Compute Platform' section, which includes a sub-section '3.1. NSF NCAR' and a dropdown menu for 'Instructions'. The text describes the need for greater computing resources for simulations like WRF and mentions that I-WRF can run on user workstations or select HPC platforms. Below this, the '3.2. Jetstream2' section is visible, followed by another 'Instructions' dropdown. The bottom of the page shows the start of the '3.3. Red Cloud' section.

*Public I-WRF documentation website  
using Github and ReadTheDocs*

## Technical Organization via GitHub Projects

- Source code and documentation is stored in a GitHub repository
- GitHub issues are used to organize tasks
- Custom labels can be applied to issues to help categorize them
- GitHub Projects support multiple views of issues to track progress
- Views can include filtering by labels to view a subset of issues
- Views can be customized to show status, assignees, associated pull requests, etc.

# I-WRF GitHub Project Board – All Deliverables View

- Created an issue for each project deliverable
- Applied label **type: deliverable** to each issue
- Filtered issues with label **type: deliverable**

The screenshot displays the GitHub Project Board for the 'I-WRF Development' project. The board is filtered by the label 'type: deliverable', showing 26 issues. The issues are organized into columns based on their status: 'In progress', 'Done', 'Pending', and 'Ready'. Each issue entry includes a title, a status label, and a list of labels. The labels include 'deliverable 1.1', 'deliverable 1.2', 'deliverable 2.1', 'component: Apptainer', 'component: Docker', and 'component: plotting'.

Title	Status	Labels
1.1 Generate desired configuration #10	In progress	deliverable 1.1, type: deliverable
1.1.1 Gather requirements #11	In progress	deliverable 1.1, type: deliverable
1.1.2 Define base configuration #12	Done	deliverable 1.1, type: deliverable
1.1.3 Quality attributes #13	Done	deliverable 1.1, type: deliverable
1.2.1 Generate containers: WRF and METplus compilation #14	Done	component: Apptainer, component: Docker
1.2.2 Generate containers: Visualization component compilation #29	Done	component: plotting, deliverable 1.2, type: deliverable
1.2.3 Terraform/Kubernetes orchestration #15	Pending	deliverable 1.2, type: deliverable
1.2.4 Container refinement #16	Ready	component: Apptainer, component: Docker
1.2.5 WRF-Chem container #30	Ready	component: Apptainer, component: Docker
2.1 Use cases: Land Use Land Cover (LULC) #31	Done	deliverable 2.1, type: deliverable
2.1.1 Sensitivity to climate change #17	Done	deliverable 2.1, type: deliverable
2.1.2 LULC impact on morphology of extreme events #18	Done	deliverable 2.1, type: deliverable
2.1.3 Degree of change of societal vulnerability #19	Done	deliverable 2.1, type: deliverable



# I-WRF GitHub Project Board – X.Y Deliverable View

- Created sub-issues for specific, assignable tasks
- Applied **deliverable: X.Y** label to related issues
- Quick view of status, assignee(s), relevant pull requests, labels, and other information

The screenshot shows the GitHub Project Board for the 'I-WRF Development' project. The board is filtered by the '1.2' deliverable. A search bar at the top shows the filter 'label:"deliverable 1.2"'. The board contains 13 issues, all of which are marked as 'Done' with a green checkmark. The issues are listed in a table with columns for Title, Status, Assignees, Link, and Labels. The labels for each issue include 'component' and 'type'.

	Title	Status	Assignees	Link...	Labels
1	1.2.1 Generate containers: WRF and METplus co... #14	Done			component: Apptainer
2	Determine where to store Docker images #7	Done	georgemccabe		deliverable 1.2 type: t
3	Pass native WRF output to MET tools #5	Done	georgemccabe		deliverable 1.2 type: t
4	Build/run WRF in container using intel compiler #40	Done	hahnd	#72	component: Apptainer
5	Update container to use Intel OneAPI when WRF ... #45	Done	hahnd		component: Docker
6	Run/test METplus container using output from W... #41	Done	briannen	#62	deliverable 1.2 type: t
7	Run Hurricane Matthew test case in WRF contai... #46	Done	rcplane	#62	component: Apptainer
8	Resolve issues with permissions for files outside o... #4	Done			component: Docker
9	Update METplus use case to use MADIS2NC wra... #51	Done	georgemccabe		component: METplus
10	Obtain observation data for Hurricane Matthew t... #47	Done	jaredalee		component: METplus
11	Run Hurricane Matthew test case in METplus co... #48	Done	georgemccabe...	#53	component: METplus
12	Run Hurricane Matthew case end-to-end in WRF... #49	Done	georgemccabe		component: METplus
13	1.2.2 Generate containers: Visualization compon... #29	Done	georgemccabe	#79	component: plotting



# I-WRF GitHub Project Board – Deliverable Issue

## 1.2.1 Deliverable Issue:

- Recently, GitHub added the ability to define issue “relationship” to officially relate parent and sub-issues
- Sub-issues can now be viewed from the parent issue

The screenshot shows a GitHub issue page for the project 'I-WRF Development'. The issue title is '1.2.1 Generate containers: WRF and METplus compilation #14'. It is marked as 'Closed' and has 11 sub-issues. The issue was opened by georgemccabe on May 23, 2023. The description states: 'Create containers to run WRF and METplus. Document instructions to run basic use case.' The issue is labeled with 'component: Apptainer', 'component: Docker', 'deliverable 1.2', and 'type: deliverable'. The sub-issues listed are:

- ✓ Determine where to store Docker images #7
- ✓ Pass native WRF output to MET tools #5
- ✓ Build/run WRF in container using intel compiler #40
- ✓ Update container to use Intel OneAPI when WRF 4.5.2 is available
- ✓ Run/test METplus container using output from WRF
- ✓ Run Hurricane Matthew test case in WRF container
- Resolve issues with permissions for files outside of container
- ✓ Update METplus use case to use MADIS2NC wrapper when available

The right sidebar shows the issue's status as 'Done', size as 'X-Large', and cycle as '2023 - Year 1' (Aug 1 - Jul 31).

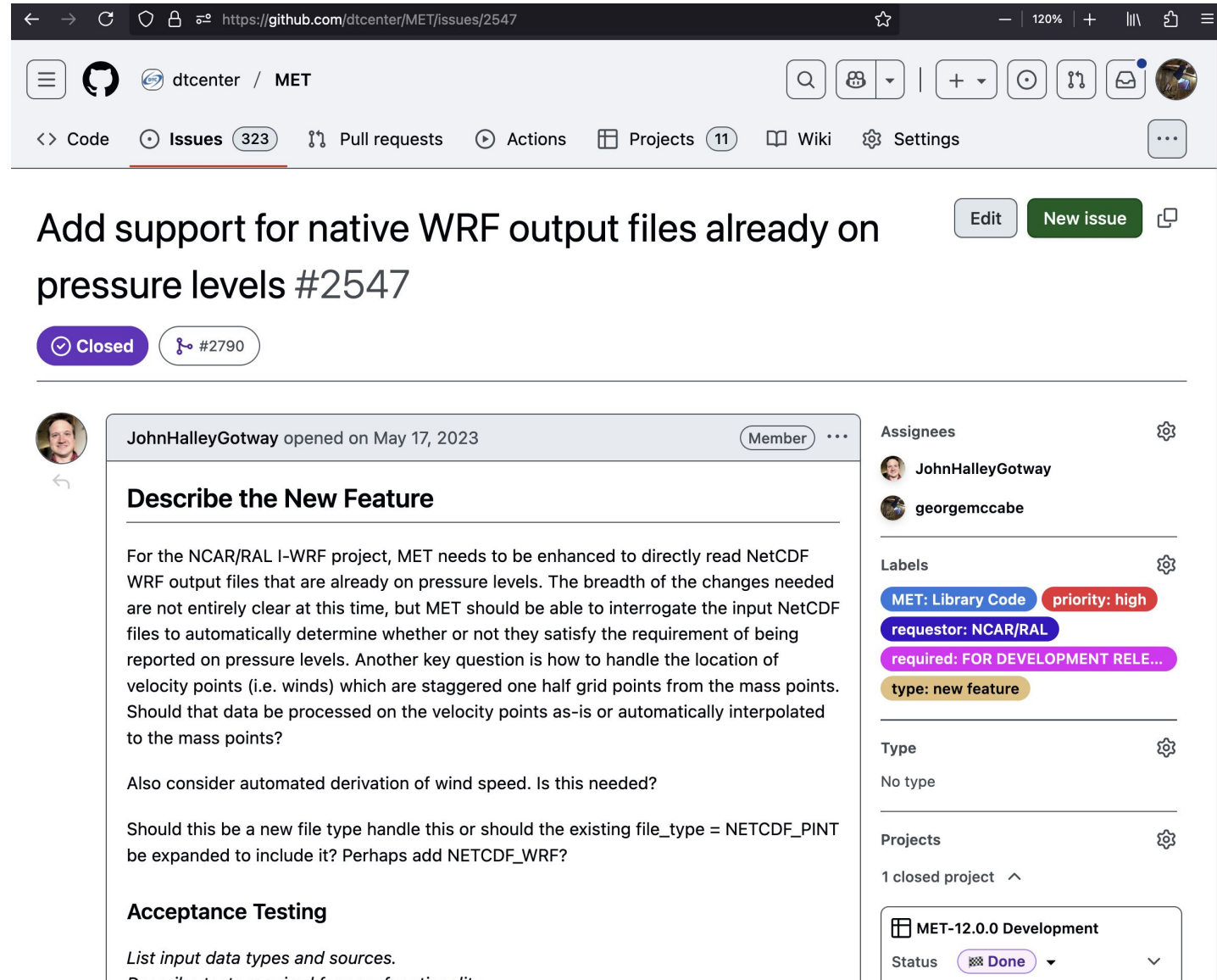
# I-WRF GitHub Project Board – Sub-Issue

- Issue details are provided using an issue template
- Add comments to provide updates and ask questions
- Can include links to other I-WRF issues or issues from external repositories
- Status of external issue can be easily viewed from I-WRF issue by mousing over link

The screenshot shows a GitHub issue page for the repository 'NCAR / i-wrf'. The issue title is 'Update METplus use case to use MADIS2NC when it is available #51'. The issue is marked as 'Closed' and has a parent issue '1.2.1 Generate containers: WRF and METplus compilation'. A comment by user 'georgemccabe' is visible, dated April 12, 2024. The comment text is: 'The MADIS2NC wrapper has not yet been created (issue [dtcenter/METplus#1514](#)). The METplus use case (#48) can still call the madis2nc MET tool by using the UserScript wrapper.' Below the comment, there is a section titled 'Describe the Task' with the text: 'When the MADIS2NC wrapper is available, update the METplus use case to call the wrapper instead of UserScript.' There is also a 'Time Estimate' section showing '~1 day' and a 'Sub-Issues' section with the text 'Consider breaking the task down into sub-issues.' and a checkbox 'Add a checkbox for each sub-issue here.' which is checked. A tooltip is visible over the link 'dtcenter/METplus#1514', showing details for issue 'New Wrapper: MADIS2NC #1514'. The tooltip includes the description 'Describe the New Feature', 'Create METplus wrapper to easily configure madis2nc MET tool.', and labels 'component: python...', 'priority: high', and 'type: new feature'. On the right side of the page, there is a 'Labels' section with labels 'component: METplus', 'deliverable 1.2', 'priority: medium', 'requestor: NCAR/RAL', and 'type: task'. There is also a 'Projects' section showing 'I-WRF Development' with a status of 'Done'. At the bottom, there is a 'Milestone' section showing 'I-WRF 1.0.0' with a due date of 'Past due by 10m 23d, 100% complete'.

# METplus Development

- MET was enhanced to support reading WRF output files directly without using a Python script
- An issue in the MET repository was created to track this work



The screenshot shows a GitHub issue page for the MET repository. The issue title is "Add support for native WRF output files already on pressure levels #2547". The issue is marked as "Closed" and has a link to #2790. The issue was opened by JohnHalleyGotway on May 17, 2023. The issue description is titled "Describe the New Feature" and contains the following text:

For the NCAR/RAL I-WRF project, MET needs to be enhanced to directly read NetCDF WRF output files that are already on pressure levels. The breadth of the changes needed are not entirely clear at this time, but MET should be able to interrogate the input NetCDF files to automatically determine whether or not they satisfy the requirement of being reported on pressure levels. Another key question is how to handle the location of velocity points (i.e. winds) which are staggered one half grid points from the mass points. Should that data be processed on the velocity points as-is or automatically interpolated to the mass points?

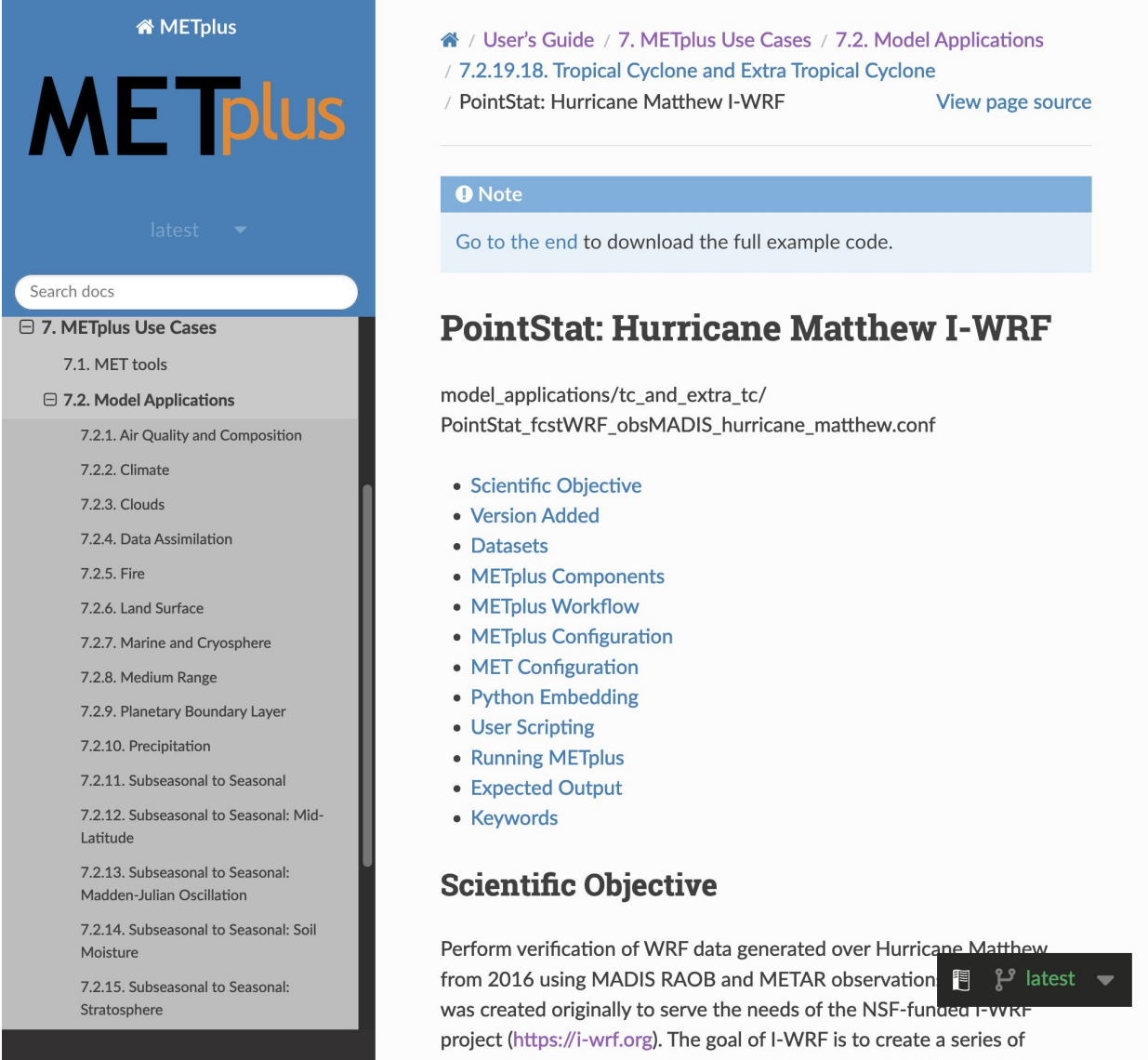
Also consider automated derivation of wind speed. Is this needed?

Should this be a new file type handle this or should the existing file\_type = NETCDF\_PINT be expanded to include it? Perhaps add NETCDF\_WRF?

The issue is labeled with "MET: Library Code", "priority: high", "requestor: NCAR/RAL", "required: FOR DEVELOPMENT RELE...", and "type: new feature". The issue is assigned to JohnHalleyGotway and georgemccabe. The issue is also linked to the project "MET-12.0.0 Development" with a status of "Done".

# METplus Use Case

- METplus portion of the I-WRF Hurricane Matthew use case was added to the METplus use cases
- Contributes example of reading WRF into MET that is useful to the community
- Increases visibility of I-WRF project



The screenshot displays the METplus documentation website. The header features the METplus logo and a navigation bar with links to the User's Guide, METplus Use Cases, and Model Applications. A sidebar on the left lists the contents of the '7.2. Model Applications' section, including various WRF model applications. The main content area shows the 'PointStat: Hurricane Matthew I-WRF' page, which includes a note about downloading example code, a list of links to related documentation, and a 'Scientific Objective' section.

**METplus**

latest

Search docs

7. METplus Use Cases

- 7.1. MET tools
- 7.2. Model Applications
  - 7.2.1. Air Quality and Composition
  - 7.2.2. Climate
  - 7.2.3. Clouds
  - 7.2.4. Data Assimilation
  - 7.2.5. Fire
  - 7.2.6. Land Surface
  - 7.2.7. Marine and Cryosphere
  - 7.2.8. Medium Range
  - 7.2.9. Planetary Boundary Layer
  - 7.2.10. Precipitation
  - 7.2.11. Subseasonal to Seasonal
  - 7.2.12. Subseasonal to Seasonal: Mid-Latitude
  - 7.2.13. Subseasonal to Seasonal: Madden-Julian Oscillation
  - 7.2.14. Subseasonal to Seasonal: Soil Moisture
  - 7.2.15. Subseasonal to Seasonal: Stratosphere

7.2. Model Applications

7.2.19.18. Tropical Cyclone and Extra Tropical Cyclone

PointStat: Hurricane Matthew I-WRF

View page source

Note


Go to the end to download the full example code.

**PointStat: Hurricane Matthew I-WRF**

model\_applications/tc\_and\_extra\_tc/  
PointStat\_fcstWRF\_obsMADIS\_hurricane\_matthew.conf

- Scientific Objective
- Version Added
- Datasets
- METplus Components
- METplus Workflow
- METplus Configuration
- MET Configuration
- Python Embedding
- User Scripting
- Running METplus
- Expected Output
- Keywords

**Scientific Objective**

Perform verification of WRF data generated over Hurricane Matthew from 2016 using MADIS RAOB and METAR observation.  latest

was created originally to serve the needs of the NSF-funded I-WRF project (<https://i-wrf.org>). The goal of I-WRF is to create a series of connected software containers to enable multi-node WRF simulations.

# I-WRF DockerHub Repositories

- ncar/iwrf : WRF Container
  - Contains software requires to run WRF
  - <https://hub.docker.com/repository/docker/ncar/iwrf>
- ncar/iwrf-metplus : METplus Container
  - Contains METplus wrappers, MET C++ executables, METplus Analysis plotting tools, and WRF-Python package
  - <https://hub.docker.com/repository/docker/ncar/iwrf-metplus>
- ncar/iwrf-data : Data Volumes
  - Contains input data used to run I-WRF use cases
  - <https://hub.docker.com/repository/docker/ncar/iwrf-data>

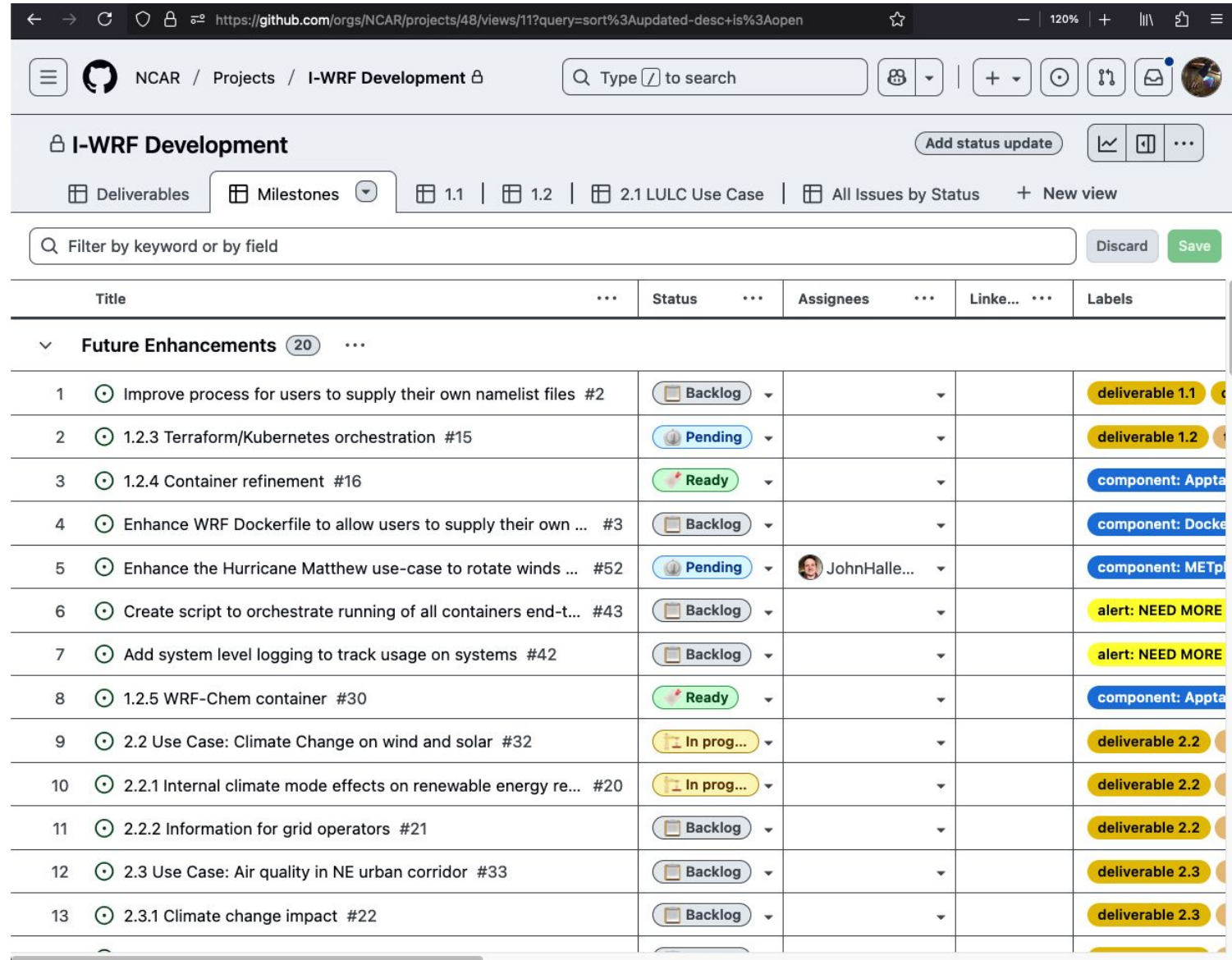
The screenshot shows the DockerHub repository page for `ncar/iwrf-metplus`. The page includes the repository name, a description of the container's purpose (MET/METplus wrappers, METplus Analysis, and wrf python), and a list of Docker commands for pushing a new tag. Below this, there are tabs for 'General', 'Tags', 'Image Management', 'Builds', 'Permissions', 'Webhooks', and 'Settings'. The 'Tags' tab is selected, showing a table of tags with columns for Tag, OS, Vulnerabilities, Health, Pulled, and Pushed. The table lists four tags: 2025-08-18, 2024-10-07, 2024-11-20, and latest. A 'DOCKER SCOUT INACTIVE' banner is visible in the top right corner of the tags section.

Tag	OS	Vulnerabilities	Health	Pulled	Pushed
2025-08-18	linux/amd64	Security unknown	N/A	less than 1 day	about 22 hours
2024-10-07	linux/amd64	Security unknown	N/A	3 months	3 months
2024-11-20	linux/amd64	Security unknown	N/A	4 days	3 months
latest	linux/amd64	Security unknown	N/A	4 days	3 months



# I-WRF GitHub Project Board – Future Enhancements

- Milestones View groups issues by milestone
- Currently all remaining issues are listed under Future Enhancements
- Remaining issues will be moved to the appropriate milestone and assigned to I-WRF team members



The screenshot shows the GitHub Project Board for the 'I-WRF Development' project. The 'Milestones' tab is selected, and the 'Future Enhancements' milestone is expanded, showing 20 issues. The issues are listed in a table with columns for Title, Status, Assignees, Links, and Labels.

Title	Status	Assignees	Links	Labels
1 Improve process for users to supply their own namelist files #2	Backlog			deliverable 1.1
2 1.2.3 Terraform/Kubernetes orchestration #15	Pending			deliverable 1.2
3 1.2.4 Container refinement #16	Ready			component: Appt
4 Enhance WRF Dockerfile to allow users to supply their own ... #3	Backlog			component: Docker
5 Enhance the Hurricane Matthew use-case to rotate winds ... #52	Pending	JohnHalle...		component: METP
6 Create script to orchestrate running of all containers end-t... #43	Backlog			alert: NEED MORE
7 Add system level logging to track usage on systems #42	Backlog			alert: NEED MORE
8 1.2.5 WRF-Chem container #30	Ready			component: Appt
9 2.2 Use Case: Climate Change on wind and solar #32	In prog...			deliverable 2.2
10 2.2.1 Internal climate mode effects on renewable energy re... #20	In prog...			deliverable 2.2
11 2.2.2 Information for grid operators #21	Backlog			deliverable 2.2
12 2.3 Use Case: Air quality in NE urban corridor #33	Backlog			deliverable 2.3
13 2.3.1 Climate change impact #22	Backlog			deliverable 2.3

Questions?