JUPITER

# Cloud-based WRF Downscaling

## Simulations at Scale using Community Reanalysis and Climate Datasets

Luke Madaus -- 26 June 2018
luke.madaus@jupiterintel.com

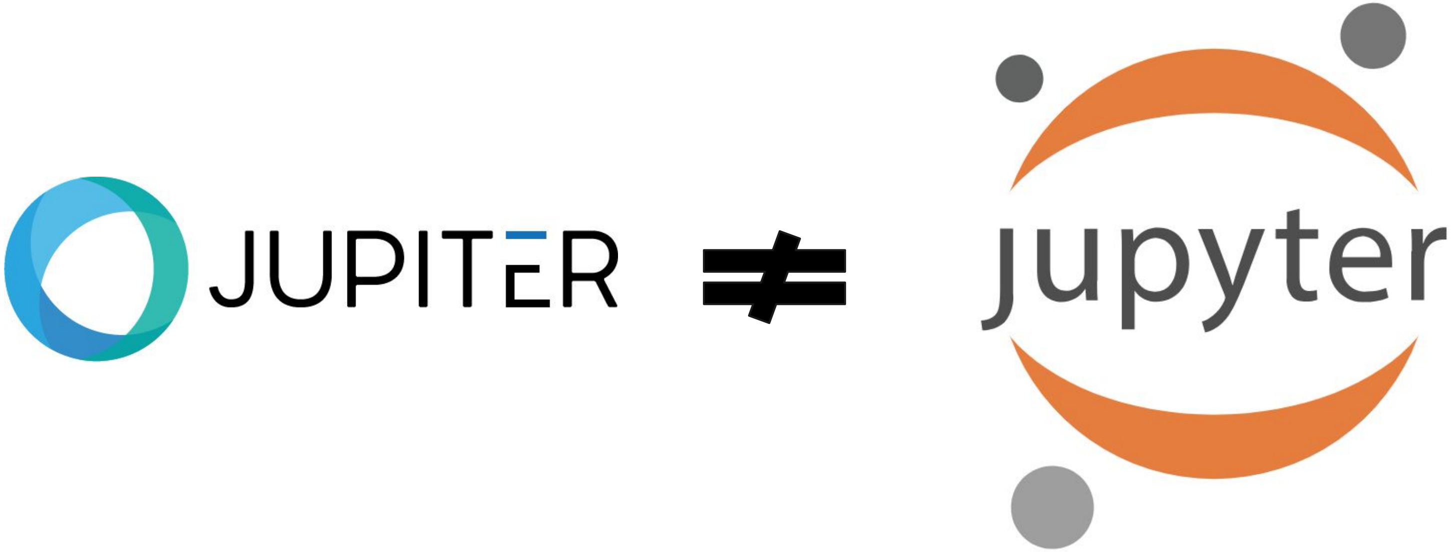
2018 Unidata Users Workshop

# Outline

- What is Jupiter?
- Challenges to running WRF at scale
- Jupiter’s approach to this problem
  - Leveraging community datasets
  - Containerization
  - Distributed data-proximate analysis
- (Brief) demonstration
- Example use cases
- Looking ahead to broader access
- Questions for the group

# What is Jupiter?

# What is Jupiter?

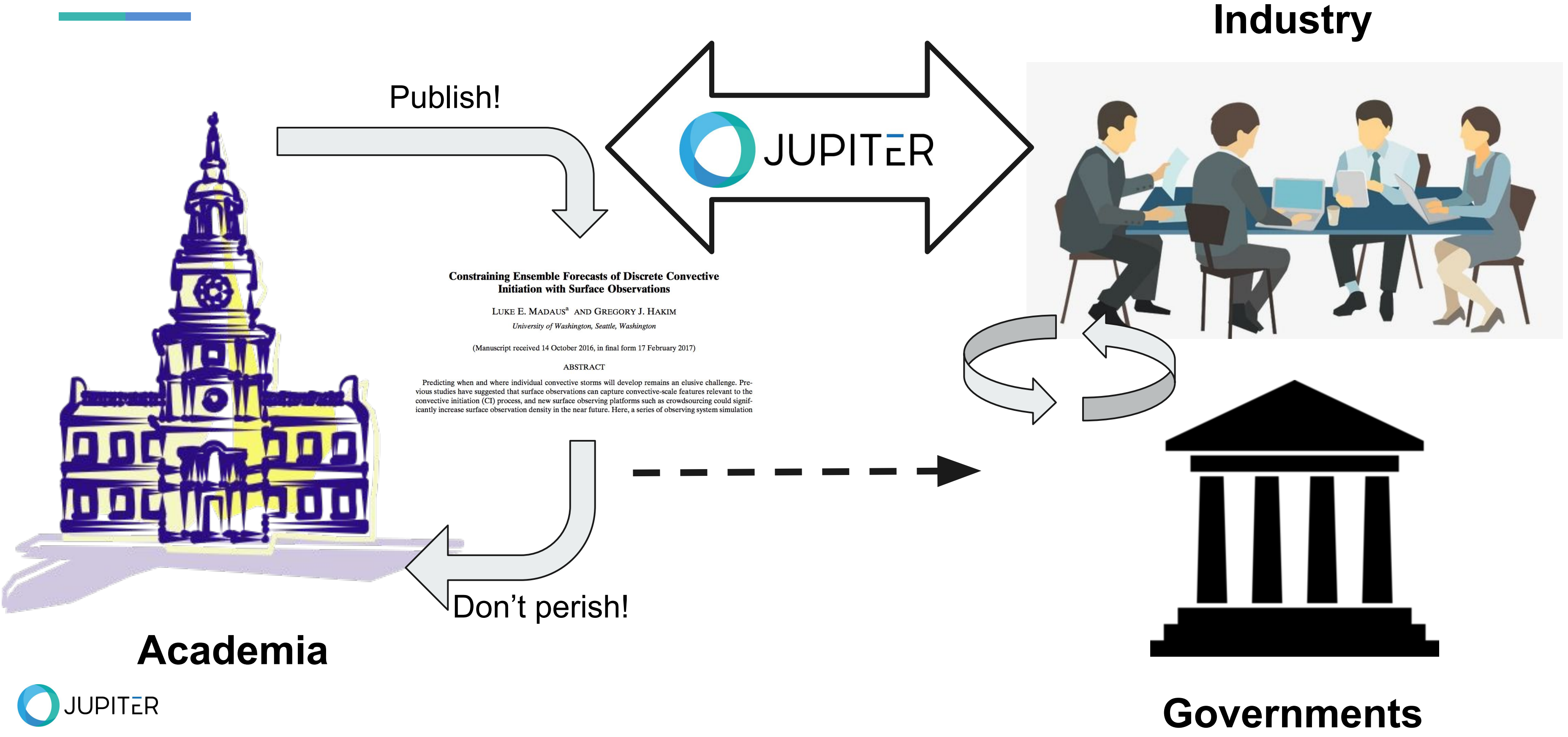


# What is Jupiter?

JUPITER =

**Translating the Latest Climate Science to the Needs of Diverse Users**

# What is Jupiter?



Publish!

**Constraining Ensemble Forecasts of Discrete Convective Initiation with Surface Observations**

Luke E. Madaus[a] and Gregory J. Hakim

*University of Washington, Seattle, Washington*

(Manuscript received 14 October 2016, in final form 17 February 2017)

ABSTRACT

Predicting when and where individual convective storms will develop remains an elusive challenge. Previous studies have suggested that surface observations can capture convective-scale features relevant to the convective initiation (CI) process, and new surface observing platforms such as crowdsourcing could significantly increase surface observation density in the near future. Here, a series of observing system simulation

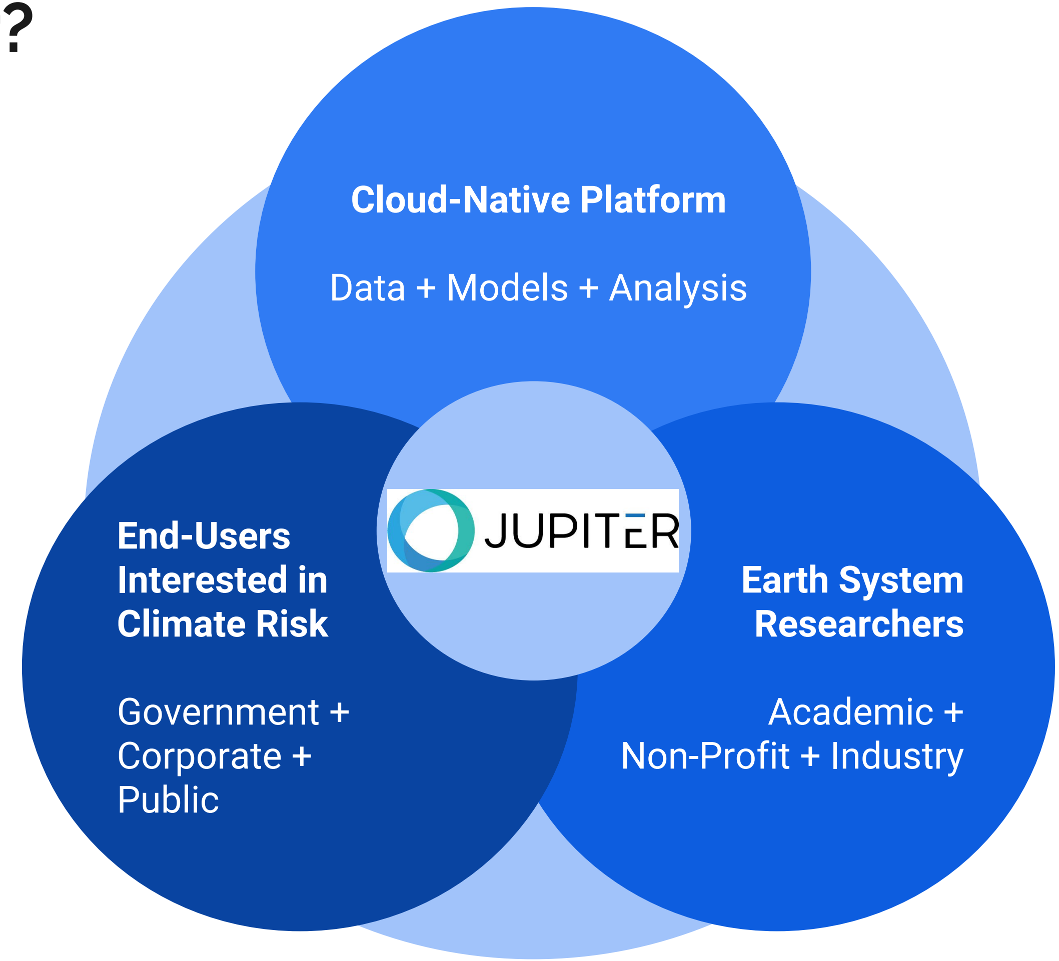
Don't perish!

**Academia**

**Industry**

**Governments**

# What is Jupiter?



**Cloud-Native Platform**

Data + Models + Analysis

**End-Users Interested in Climate Risk**

Government + Corporate + Public

**Earth System Researchers**

Academic + Non-Profit + Industry

# What is Jupiter?

**FloodScore**
(Operations and Planning)

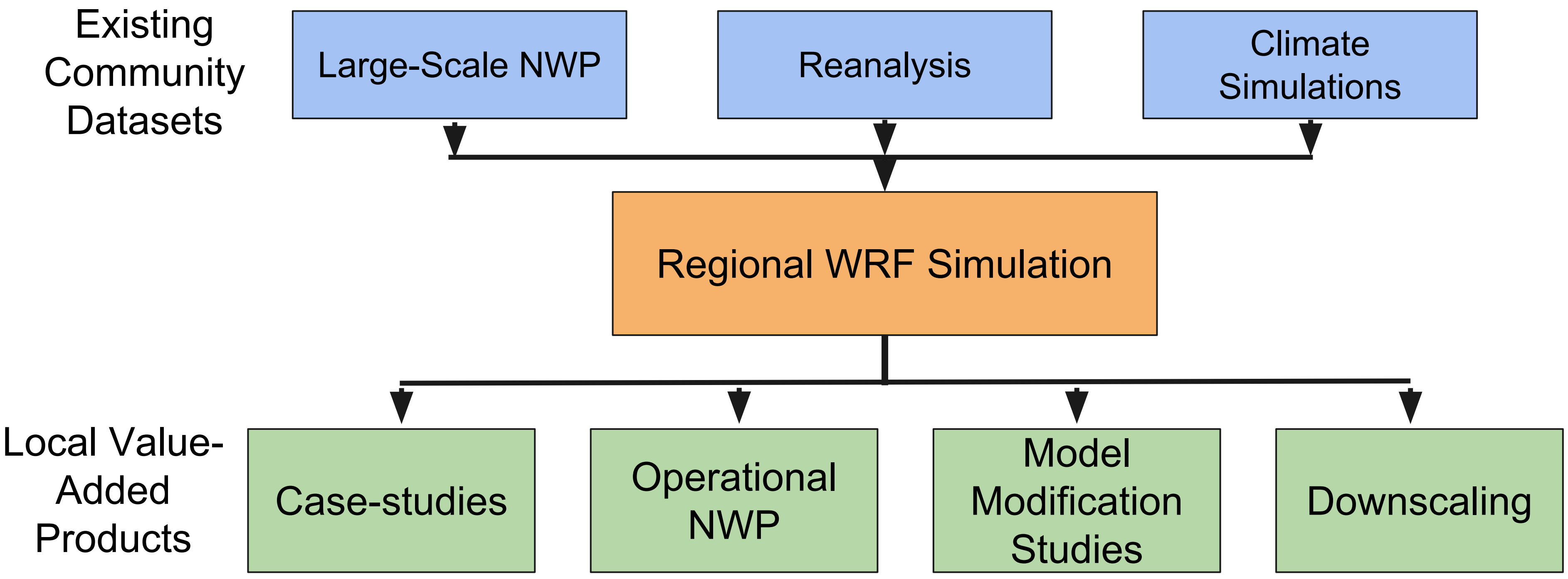
**HeatScore**

(Downscaling)

How are ***hyper-local-scale*** risks for flood and extreme heat hazards changing on ***daily to decadal timescales***?

# Regional WRF Simulations on the cloud -- Challenges

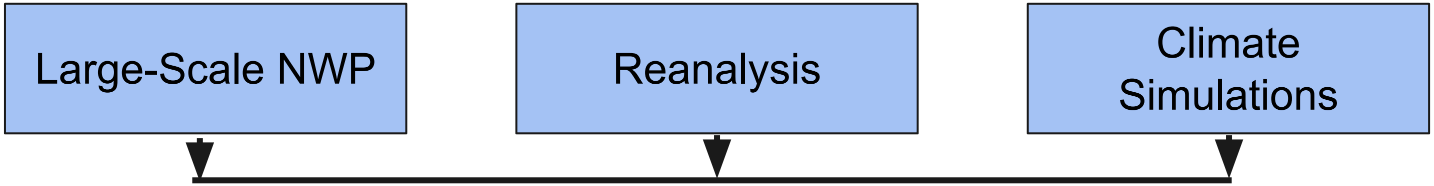
# Regional WRF Simulation



JUPITER

# Challenges

Existing Community Datasets



Large-Scale NWP

Reanalysis

Climate Simulations

**Large files designed for bulk downloads!**

Current Solution: Have to download everything even if you just need a handful of dates/variables

**Distributed in netCDF, WRF designed for GRIB!**

Current Solution: Complicated reformatting and repackaging

**Variables not in the right format!**

Current Solution: Have to do some sort of calculations or manipulations on data to make it usable for WRF

# Challenges

Regional WRF Simulation

**How do I make WRF work on my system?**

Current Solution: While WRF and supporting libraries have gotten easier to compile and install, still a challenge

**I want to simply swap out another version of my code?**

Current Solution: Changing dynamic links, keep track of pointers, making sure things aren't overwritten...

**I need to move my work to a new cluster. How?**

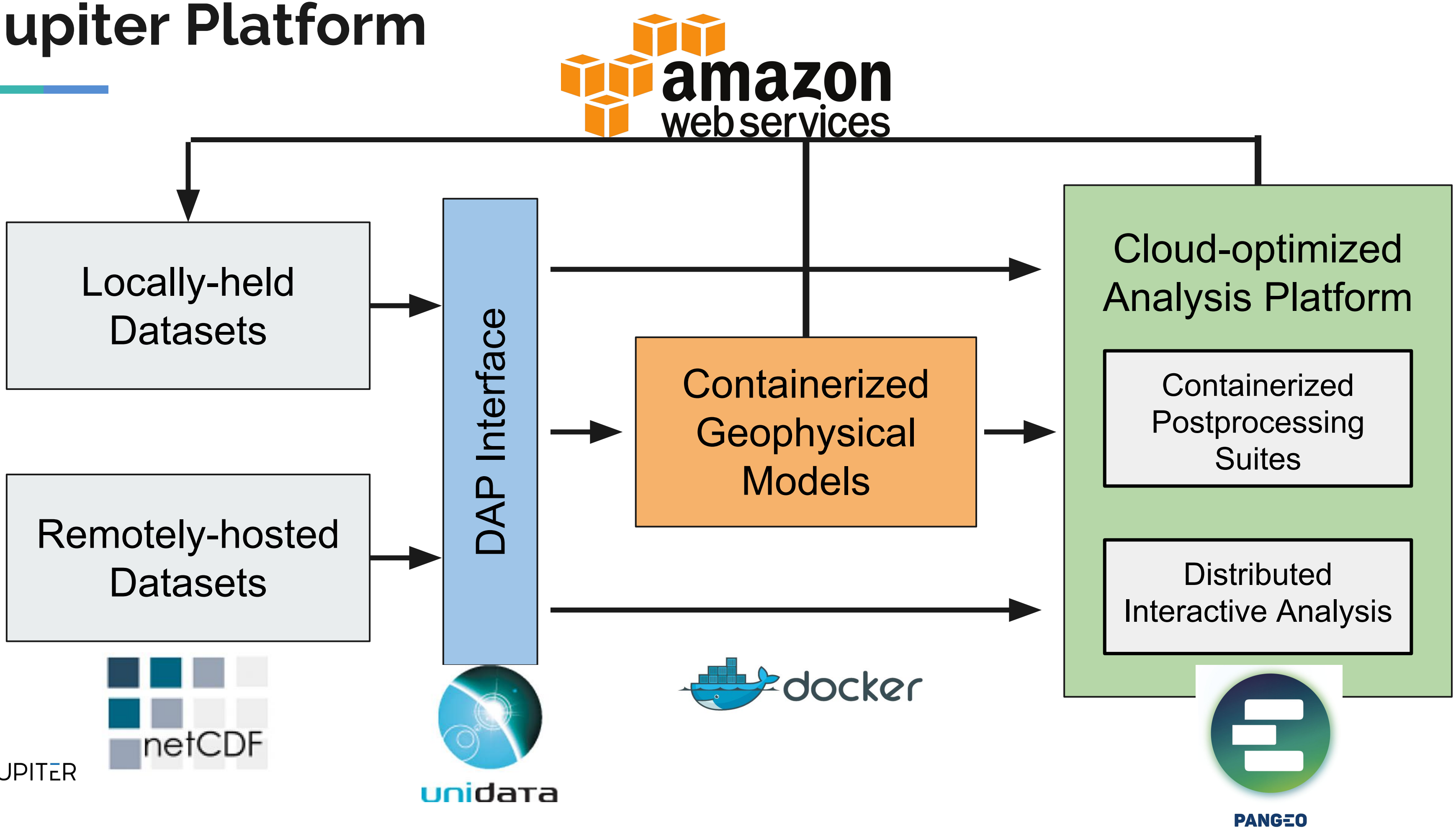
Current Solution: Reinstalling libraries, recompiling, running baseline checks again...

**How do I share all the data I'm generating?**

Current Solution: Downloading to local machine, setting up own file server...

# Regional WRF Simulations on the cloud -- Jupiter’s Approach

# Jupiter Platform



Locally-held Datasets

Remotely-hosted Datasets

DAP Interface

Containerized Geophysical Models

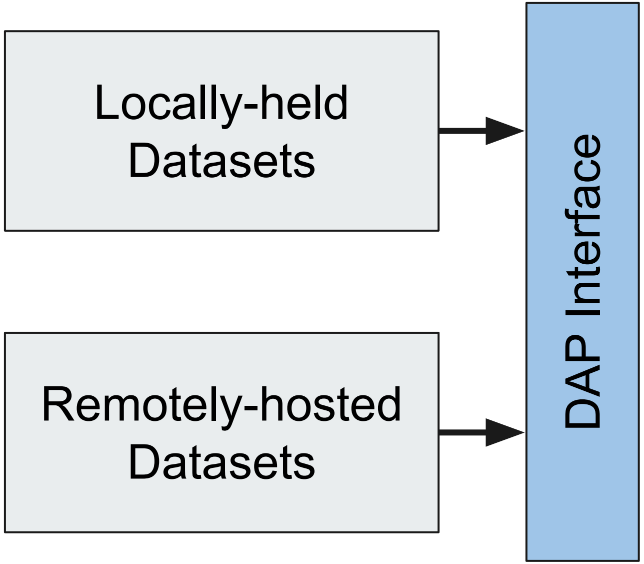
Cloud-optimized Analysis Platform

Containerized Postprocessing Suites

Distributed Interactive Analysis

JUPITER

# Jupiter Platform



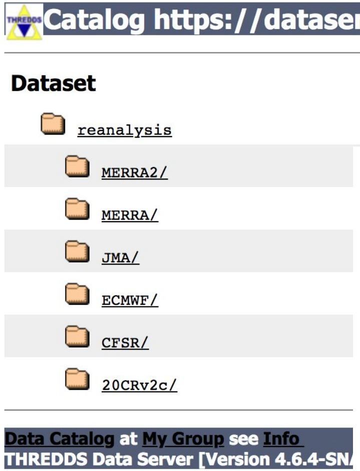
## The Data

- netCDF format as current standard
- Increasing number of weather and climate datasets hosted through streamable platforms (e.g., THREDDS)
- On-the-fly subsetting, minimize download size
- Jupiter exploring more cloud-optimized data formats and hosting methods (e.g. Zarr...more on this at the end)

# DAP-Fronted Community Datasets

NOAA-ESRL

NASA -- CREATE-IP
Reanalysis Datasets (Potter et al. 2018[1])

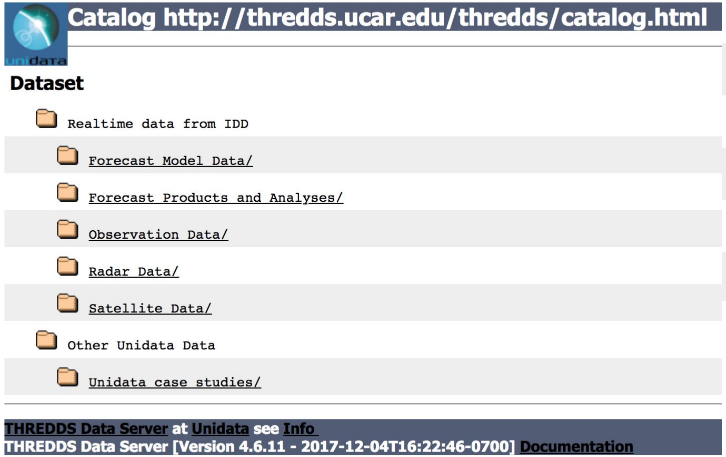


Catalog https://dataserver.nccs.nasa.gov/thredds/catalog/

Dataset

- reanalysis
  - MERRA2/
  - MERRA/
  - JMA/
  - ECMWF/
  - CFSR/
  - 20CRv2c/

Data Catalog at My Group see Info
THREDDS Data Server [Version 4.6.4-SN

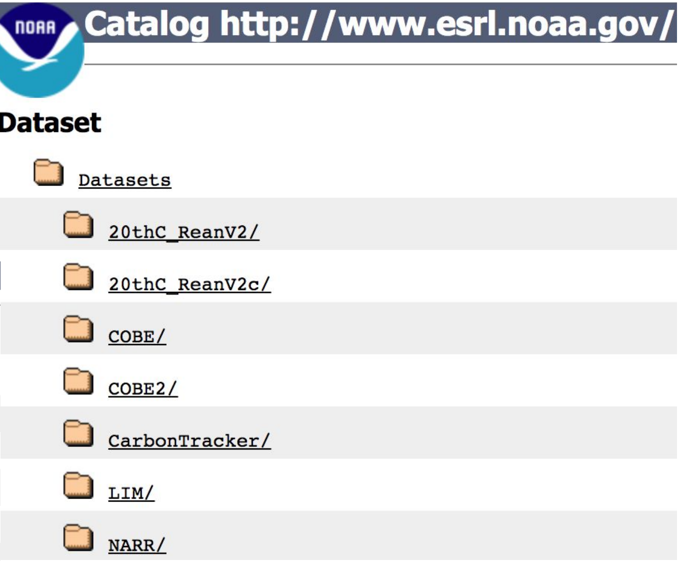


Catalog http://thredds.ucar.edu/thredds/catalog.html

Dataset

- Realtime data from IDD
  - Forecast Model Data/
  - Forecast Products and Analyses/
  - Observation Data/
  - Radar Data/
  - Satellite Data/
- Other Unidata Data
  - Unidata case studies/

THREDDS Data Server at Unidata see Info
THREDDS Data Server [Version 4.6.11 - 2017-12-04T16:22:46-0700] Documentation



Catalog http://www.esrl.noaa.gov/

Dataset

- Datasets
  - 20thC_ReanV2/
  - 20thC_ReanV2c/
  - COBE/
  - COBE2/
  - CarbonTracker/
  - LIM/
  - NARR/

Unidata

Many more...

JUPITER

[1] - Potter, G.L., L. Carriere, J. Hertz, M. Bosilovich, D. Duffy, T. Lee, and D.N. Williams, 2018: Enabling Reanalysis Research Using the Collaborative Reanalysis Technical Environment (CREATE). *Bull. Amer. Meteor. Soc.*, **99**, 677–687, https://doi.org/10.1175/BAMS-D-17-0174.1

# Jupiter Platform

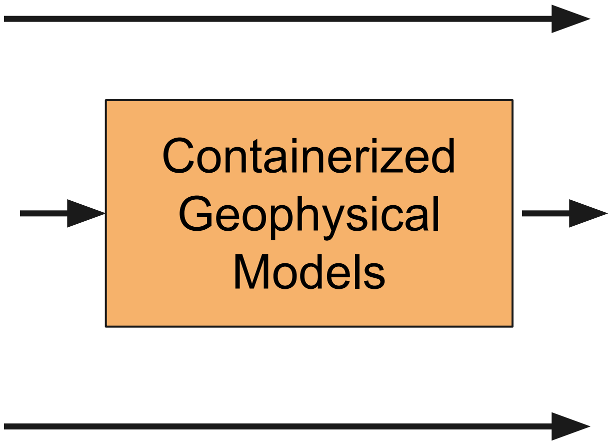
## The Models

- NSF BigWeatherWeb: Containerization allows confident deployment of models at scale (Hacker et al. 2017[1])

- Preprocessing automated -- ***direct from netCDF to WRF***

- Batch and on-demand simulation for research and real-time forecasts

- Pushing computational limits of current cloud compute architectures

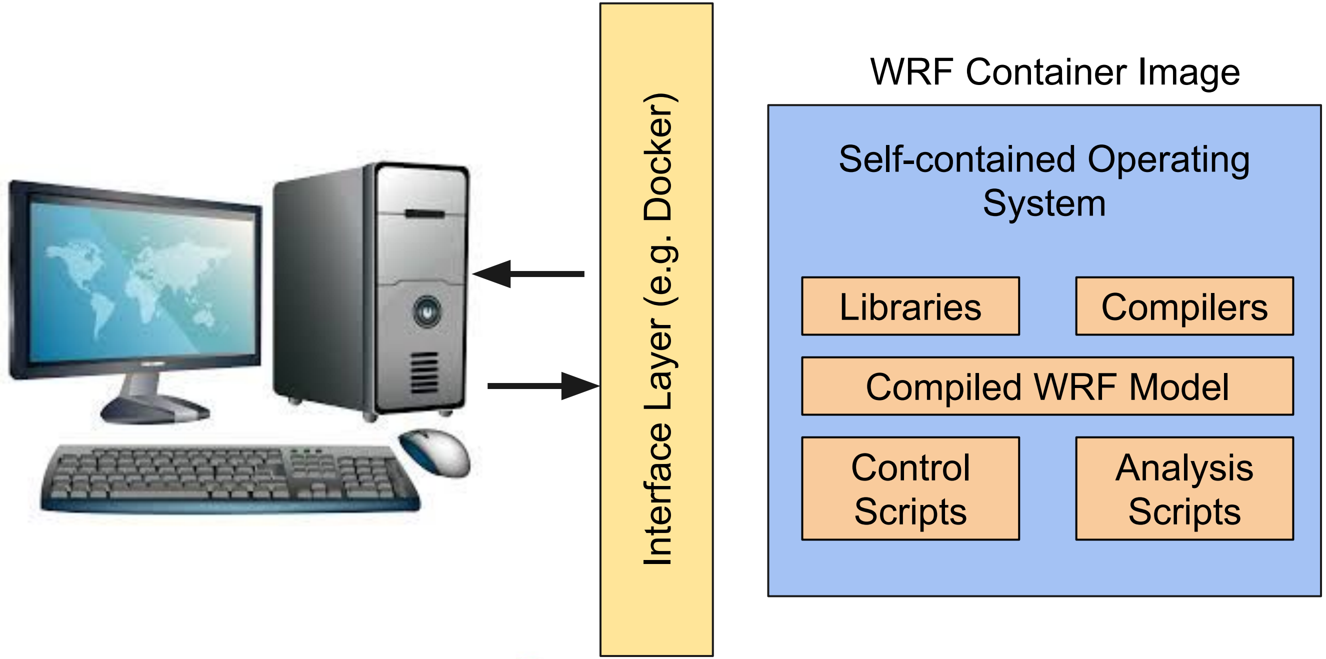


amazon web services

JUPITER

[1] - Hacker, J.P., J. Exby, D. Gill, I. Jimenez, C. Maltzahn, T. See, G. Mullendore, and K. Fossell, 2017: A Containerized Mesoscale Model and Analysis Toolkit to Accelerate Classroom Learning, Collaborative Research, and Uncertainty Quantification. *Bull. Amer. Meteor. Soc.*, **98**, 1129–1138, https://doi.org/10.1175/BAMS-D-15-00255.1

# What is containerization?



Interface Layer (e.g. Docker)

WRF Container Image

Self-contained Operating System

Libraries

Compilers

Compiled WRF Model

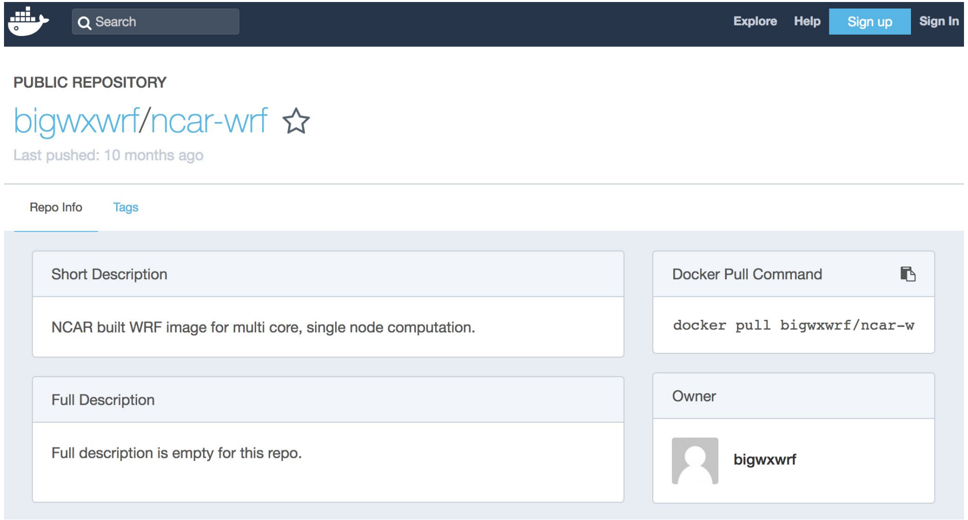
Control Scripts

Analysis Scripts

A portable, self-contained, only what is needed package that can perform some function on ***any machine*** running an ***interface layer***.

# What is containerization?

Pre-built container images can be “pulled” from online repositories



PUBLIC REPOSITORY

bigwxwrf/ncar-wrf ☆

Last pushed: 10 months ago

Short Description

NCAR built WRF image for multi core, single node computation.

Docker Pull Command

```
docker pull bigwxwrf/ncar-w
```

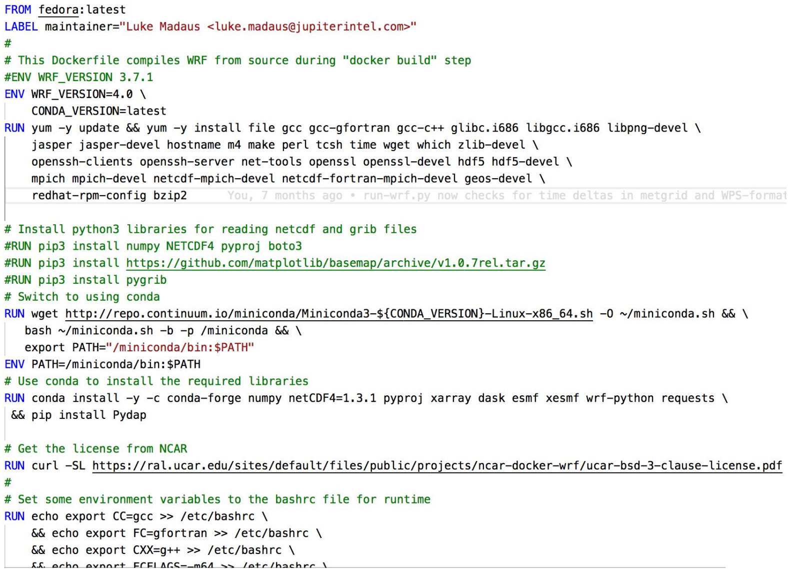
Full Description

Full description is empty for this repo.

Owner

bigwxwrf

Or, a single text file that contains a “recipe” for building the container can be downloaded for local builds



```
FROM fedora:latest
LABEL maintainer="Luke Madaus <luke.madaus@jupiterintel.com>"
#
# This Dockerfile compiles WRF from source during "docker build" step
#ENV WRF_VERSION 3.7.1
ENV WRF_VERSION=4.0 \
    CONDA_VERSION=latest
RUN yum -y update && yum -y install file gcc gcc-gfortran gcc-c++ glibc.i686 libgcc.i686 libpng-devel \
    jasper jasper-devel hostname m4 make perl tcsh time wget which zlib-devel \
    openssh-clients openssh-server net-tools openssl openssl-devel hdf5 hdf5-devel \
    mpich mpich-devel netcdf-mpich-devel netcdf-fortran-mpich-devel geos-devel \
    redhat-rpm-config bzip2       You, 7 months ago • run-wrf.py now checks for time deltas in metgrid and WPS-format

# Install python3 libraries for reading netcdf and grib files
#RUN pip3 install numpy NETCDF4 pyproj boto3
#RUN pip3 install https://github.com/matplotlib/basemap/archive/v1.0.7rel.tar.gz
#RUN pip3 install pygrib
# Switch to using conda
RUN wget http://repo.continuum.io/miniconda/Miniconda3-${CONDA_VERSION}-Linux-x86_64.sh -O ~/miniconda.sh && \
   bash ~/miniconda.sh -b -p /miniconda && \
   export PATH="/miniconda/bin:$PATH"
ENV PATH=/miniconda/bin:$PATH
# Use conda to install the required libraries
RUN conda install -y -c conda-forge numpy netCDF4=1.3.1 pyproj xarray dask esmf xesmf wrf-python requests \
 && pip install Pydap

# Get the license from NCAR
RUN curl -SL https://ral.ucar.edu/sites/default/files/public/projects/ncar-docker-wrf/ucar-bsd-3-clause-license.pdf
#
# Set some environment variables to the bashrc file for runtime
RUN echo export CC=gcc >> /etc/bashrc \
    && echo export FC=gfortran >> /etc/bashrc \
    && echo export CXX=g++ >> /etc/bashrc \
    && echo export FCFLAGS=-m64 >> /etc/bashrc \
```

# Scalable WRF Simulations in <10 lines of Python

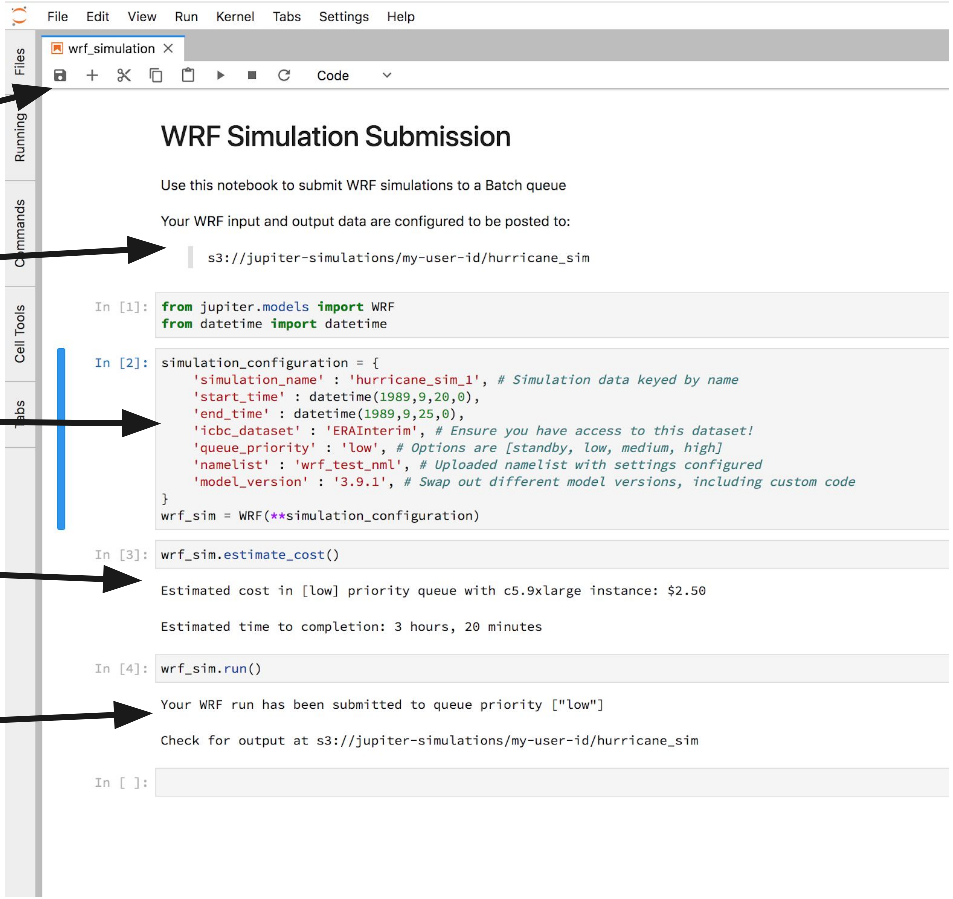
Python-based interface to platform

Cloud-based storage keeping simulation data organized and sharable

Simulation “job” options self-contained in Python object

Explore costs of running simulation with various cloud options

Command-line submission of configured jobs to cloud compute



## WRF Simulation Submission

Use this notebook to submit WRF simulations to a Batch queue

Your WRF input and output data are configured to be posted to:

```
s3://jupiter-simulations/my-user-id/hurricane_sim
```

```
In [1]: from jupiter.models import WRF
        from datetime import datetime
```

```
In [2]: simulation_configuration = {
            'simulation_name' : 'hurricane_sim_1', # Simulation data keyed by name
            'start_time' : datetime(1989,9,20,0),
            'end_time' : datetime(1989,9,25,0),
            'icbc_dataset' : 'ERAInterim', # Ensure you have access to this dataset!
            'queue_priority' : 'low', # Options are [standby, low, medium, high]
            'namelist' : 'wrf_test_nml', # Uploaded namelist with settings configured
            'model_version' : '3.9.1', # Swap out different model versions, including custom code
        }
        wrf_sim = WRF(**simulation_configuration)
```

```
In [3]: wrf_sim.estimate_cost()

        Estimated cost in [low] priority queue with c5.9xlarge instance: $2.50

        Estimated time to completion: 3 hours, 20 minutes
```

```
In [4]: wrf_sim.run()

        Your WRF run has been submitted to queue priority ["low"]

        Check for output at s3://jupiter-simulations/my-user-id/hurricane_sim
```

```
In [ ]:
```

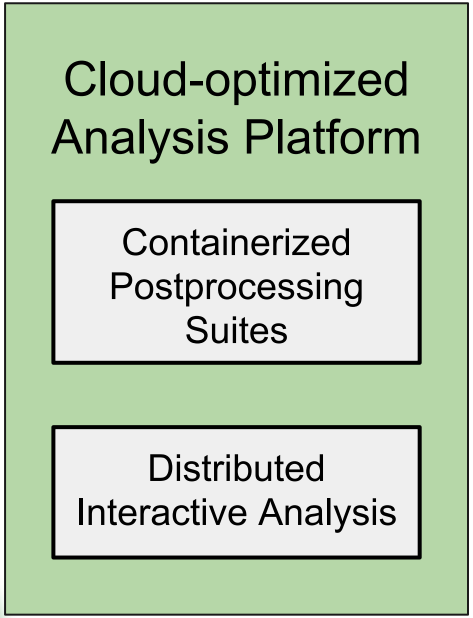
JUPITER

# Demonstration

# Jupiter Platform

## The Analysis

- Containerization allows standardized post-processing or analysis chains to be deployed
- Distributed analysis of large datasets through interactive tools (Pangeo stack)
- Allow end-users to dynamically query data for questions of interest


Cloud-optimized Analysis Platform

Containerized Postprocessing Suites

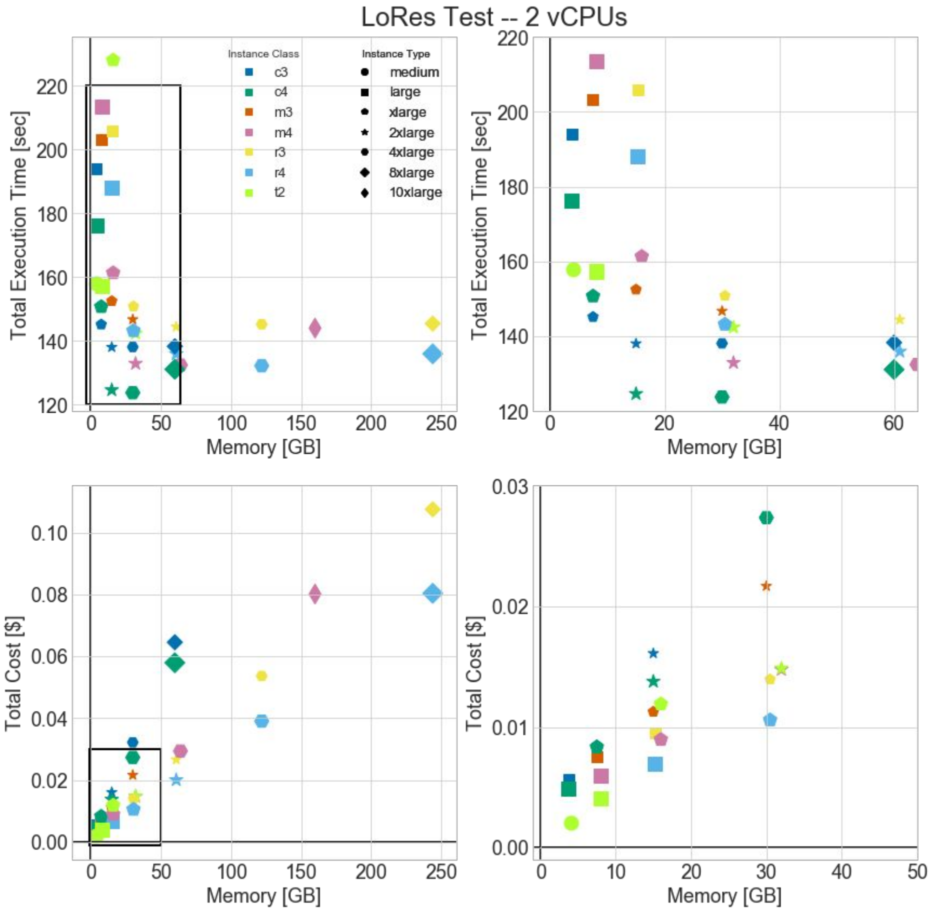
Distributed Interactive Analysis

PANGEO

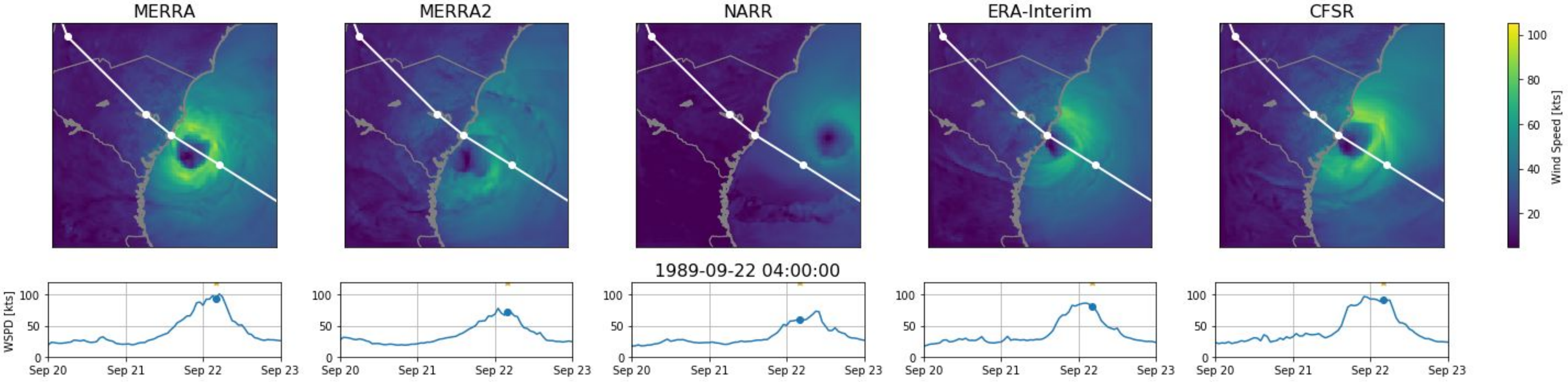
# Example Use Cases

# Example Uses

Extensive WRF benchmarking tests across a variety of cloud resource configurations



# Example Uses

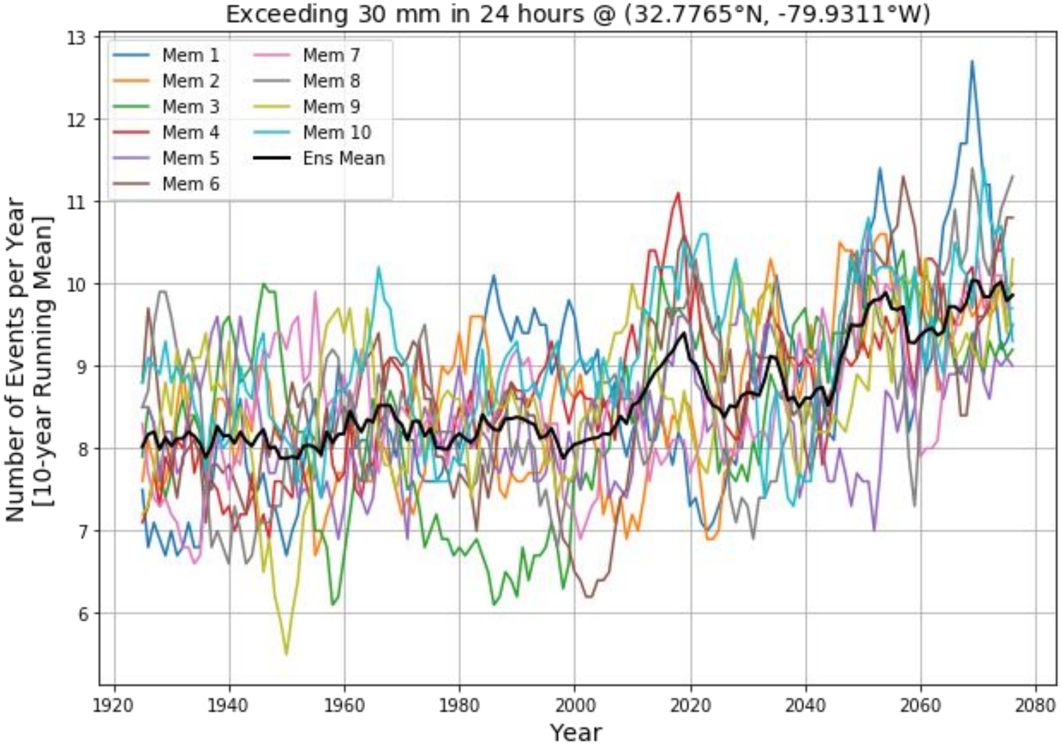


**Hurricane Hugo (1989) from the perspective of multiple reanalyses**

- None of this reanalysis data held locally
- Total pre-processing time: 1 minute per simulation (in parallel)
- WRF execution: 5 day simulation, nesting down to 3 km, 1hr 30min (in parallel)
- Conception to execution -> about 10 lines of Python on Jupiter platform

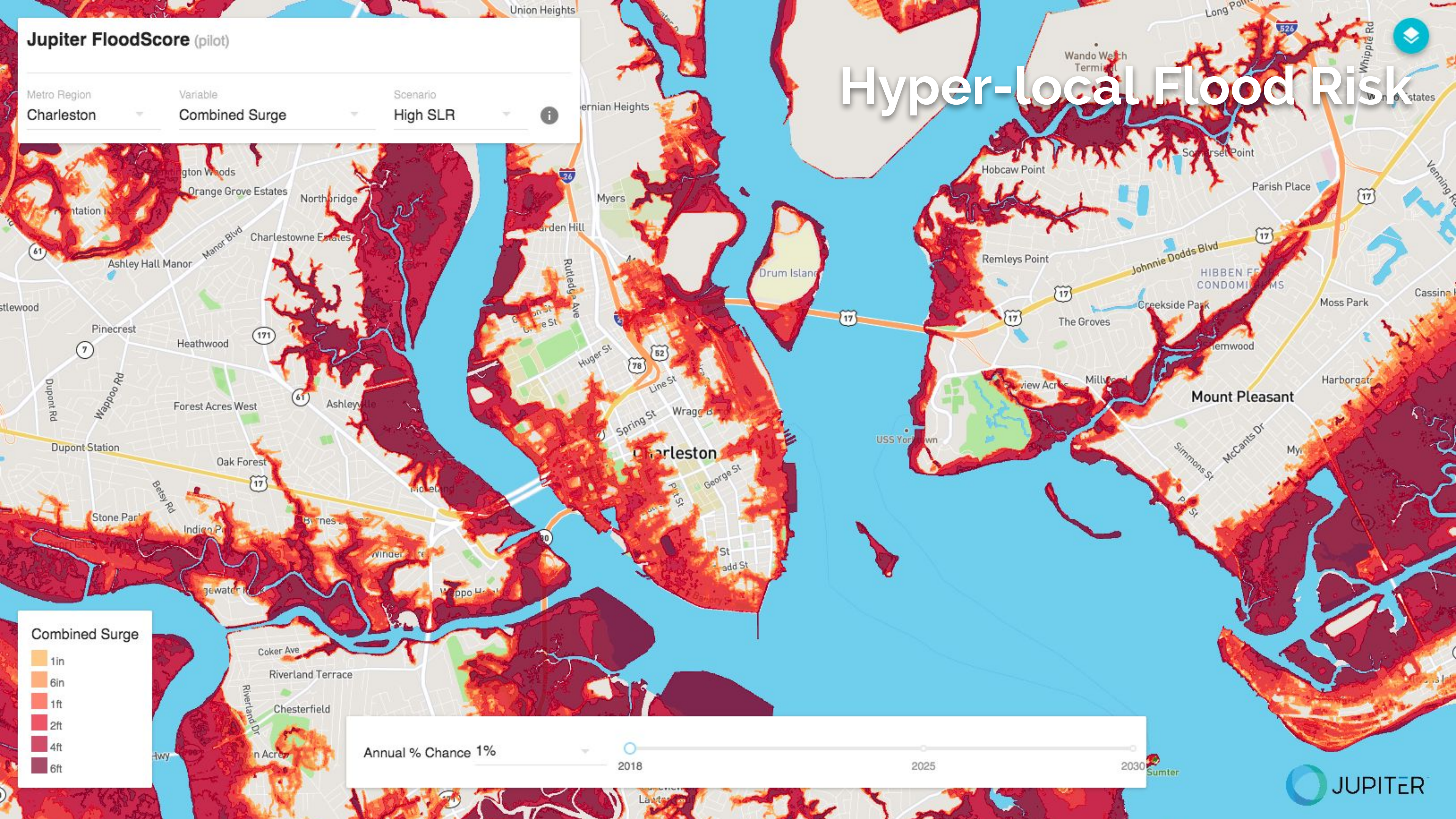
# Example Uses

- Dynamical downscaling simulations to project extreme precipitation in Charleston, SC using 10 member of the CESM-LENS
- Thousands of simulations of extreme precipitation events; completed in one weekend



JUPITER

# Hyper-local Flood Risk



**Jupiter FloodScore** (pilot)

Metro Region: Charleston

Variable: Combined Surge

Scenario: High SLR

Combined Surge

- 🟧 1in
- 🟧 6in
- 🟥 1ft
- 🟥 2ft
- 🟥 4ft
- 🟥 6ft

Annual % Chance: 1%

2018 — 2025 — 2030

# Looking Ahead to Broader Access

# BigWeatherWeb

BIG WEATHER WEB

A Common and Sustainable Big Data Infrastructure in Support of Weather Prediction Research and Education in Universities

NSF Award #1450488
Overview at:
http://bigweatherweb.org/Big_Weather_Web/Home/Home.html

BAMS Article: **A Containerized Mesoscale Model and Analysis Toolkit to Accelerate Classroom Learning, Collaborative Research, and Uncertainty Quantification**

Containerized WRF available!
https://github.com/NCAR/container-wrf
https://hub.docker.com/r/bigwxwrf/ncar-wrf/

Coming soon: Code for generating WRF initial conditions from arbitrary netCDF files

# Working on the Jupiter Platform

- Jupiter Containerized WRF targeted for Autumn 2018 release (GitHub)

- Jupiter platform access on a project-by-project basis
    - Let me know if you think you have a good use case!

# Questions for this group

1. Cloud-optimized netCDF
   a. An ever-growing need -- case for it is only strengthening
   b. Is this the way to go?
   c. How easy would it be?
2. Containerized WRF simulations
   a. Could you envision using this given your current compute platforms/resources?
   b. What features would you find useful in a standalone, containerized WRF?
3. Cloud-hosted datasets and data-proximate workflows
   a. What would it take to motivate you to move some (or all) of your work to the cloud?

luke.madaus@jupiterintel.com | @lmadaus | http://www.jupiterintel.com