Overview of the DTC's containerized end-to-end Numerical Weather Prediction (NWP) system on the cloud

Mike Kavulich[1,3], Kate Fossell[2,3], John Halley Gotway[1,3], Michelle Harrold[1,3], Jamie Wolff[1,3]

National Center for Atmospheric Research, Boulder, CO

[1]Research Applications Laboratory

[2]Mesoscale and Microscale Meteorology Laboratory

[3]Developmental Testbed Center

Developmental Testbed Center

Why are we here?

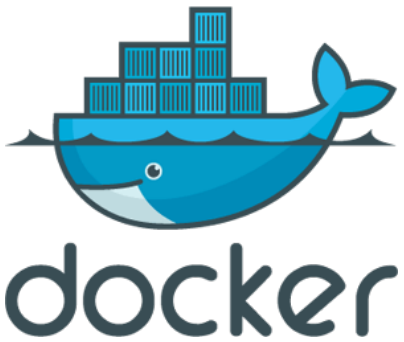
- Skills in Numerical Weather Prediction are highly desirable for many positions in earth sciences, both in academia and the private sector
- Hands on experience is an effective teaching/learning tool and resume builder
- Simple models are good, but practical applications are better
- NWP: Numerical Weather Prediction, not just model integration!
  - Pre-processing, data assimilation, model integration, post-processing, visualization, and verification

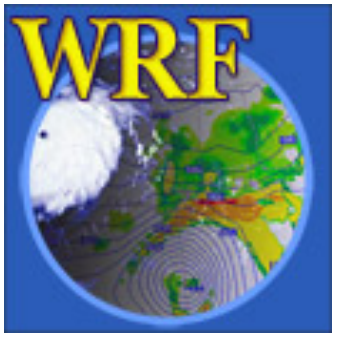
Why are we here?

- Problems
  - Compute resources on the scale needed to run dozens of simultaneous full-resolution WRF forecast are hard to justify
  - Technical skills and time needed to design, install, and troubleshoot a full NWP system can't be over-stated
- Solutions
  - We put NWP in "software containers" that allow you to move software between different machines without having to reinstall all the individual components
  - We put these containers on the cloud, because they have essentially infinitely scalable resources for relatively low cost

Technologies we leverage

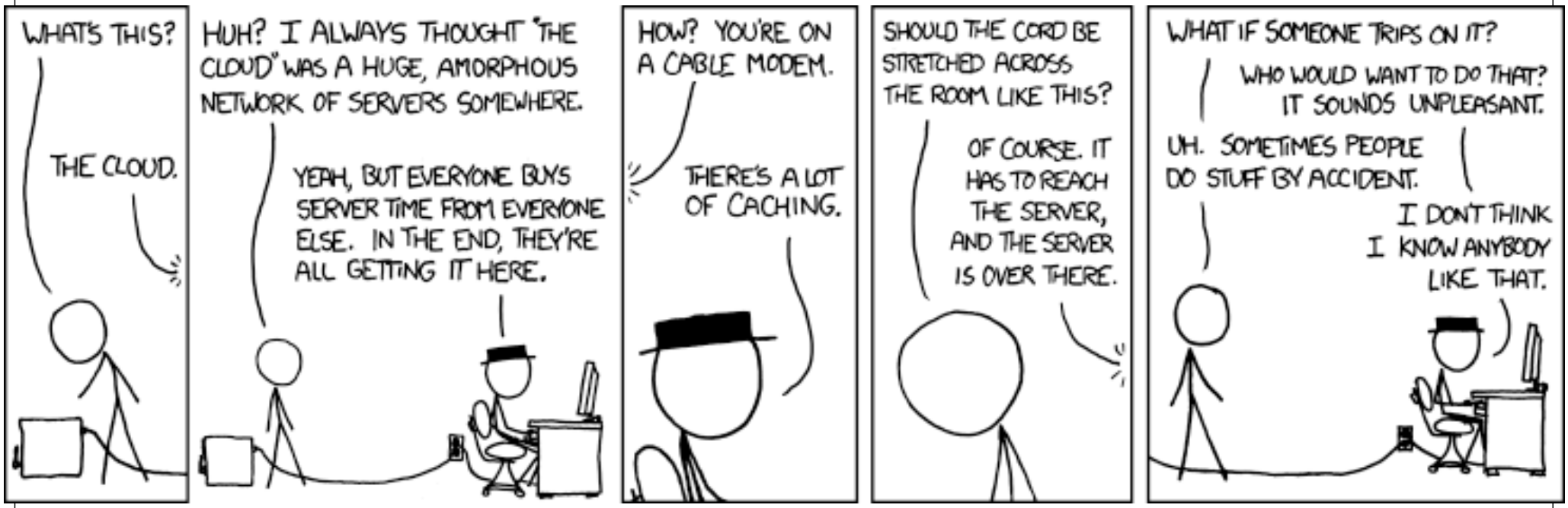
- Cloud computing
  - Amazon Web Services (AWS)
- Software containerization
  - Docker containers
- Numerical Weather Prediction (NWP) components
  - Pre-processing (WPS)
  - Data assimilation (GSI)
  - Weather model (WRF)
  - Post-processing (UPP)
  - Verification (MET)
  - Visualization (METViewer and Python)





What is cloud computing?

- The cloud is just other peoples' computers that they let you rent



https://xkcd.com/908/

Why use cloud computing?

- Computers, especially really big ones, are expensive to buy and maintain. Renting them is cheaper and easier
- Cloud provides on-demand delivery of compute power, database storage, applications, and other IT resources via the Internet.
- Access as many resources as you need almost instantly, and then shut it down when you are done; you are only charged for the compute time, storage, and bandwidth you are using.
- Amazon Web Services (AWS) specifically is a cloud services platform that owns and maintains the network-connected hardware, while you provision and use what you need via a web application.
    - Our tutorial is catered to AWS, but there are other cloud services platforms out there, such as Microsoft Azure and Google Cloud

How does AWS work?

- Log in through a normal web browser
  - Can be your personal account or set up through your institution
- In the web interface, you can browse different virtual machines
  - Many options with different amounts of memory, compute power, hardware, etc.
- When you are ready to use cloud compute resources, you will create a virtual machine (an AWS "instance") with your desired settings through the web interface
  - https://dtcenter.org/nwp-containers-online-tutorial/introduction/running-cloud
- You are charged for the time that the instance is running, whether or not you are using it
- When you are done with your instance, you should either "stop" or "terminate" it
  - A "terminated" instance will be completely deleted, with all files and history lost
  - A "stopped" instance can be restarted at a later date to continue where you left off. Stopped instances still accrue charges, but at a much lower rate than a running instance.

AWS Services & Terms

- **EC2**: Amazon Elastic Compute Cloud (EC2); provides resizable compute capacity in the cloud, includes server configuration and hosting.
  - Service to provide a virtual machine
- **Instance**: Virtual computing environments on EC2.
  - Essentially a virtual machine
  - https://aws.amazon.com/ec2/instance-types/
- **EBS**: Elastic Block Storage is block storage service that is used with EC2 instances.
  - a.k.a. your "hard drive" for your virtual machine

AWS Services & Terms

- **S3**: Amazon Simple Storage Service (S3) can be used to store and retrieve large amounts of data
  - A lot of NOAA model input data can be found on S3 storage, which can be directly imported into your instance https://registry.opendata.aws/tag/meteorological/
- **AMI**: Amazon Machine Image is a special feature that is used to create a virtual machine within the Amazon Elastic Compute Cloud ("EC2") used to deploy applications.
  - a.k.a. pre-built virtual environment
- Many, many more services and terms: https://docs.aws.amazon.com/index.html

Amazon Machine Image (AMI)

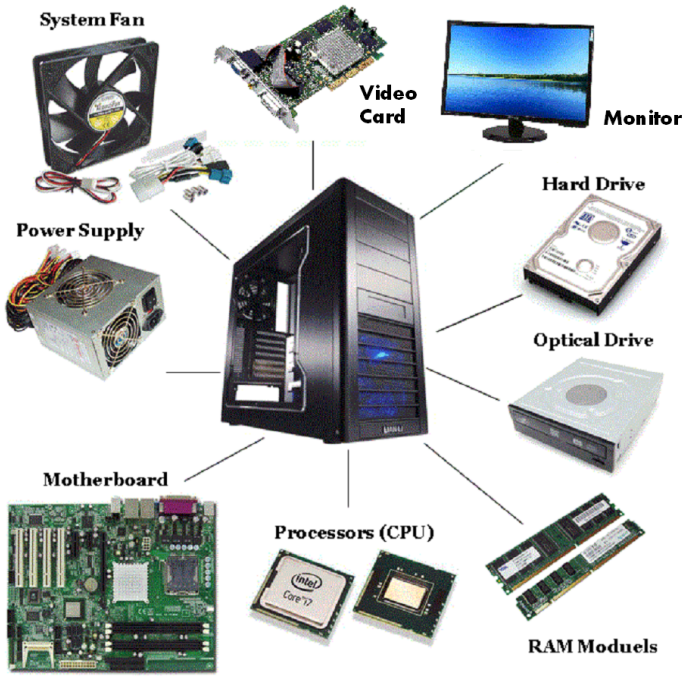
- We provide an AMI pre-loaded with hardware and software pre-configured and installed
  - EBS (60GB – enough for tutorial)
  - Model input data and observations
  - Docker
    - Docker images with NWP software already built
  - git
  - gcc
  - wgrib2
  - Python
- The tutorial is designed to either start from a fresh machine (local or cloud) or to start from the pre-configured environment of the AMI
  - If you start from a fresh machine, you will need to download and install the above prerequisites yourself

Introduction to Docker container syntax and environment

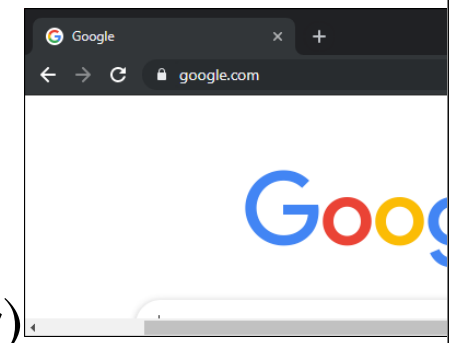
Or… “What the heck is a container?”

What is hardware? What is software?

- Hardware is the physical metal, glass, and silicon that makes up your computer

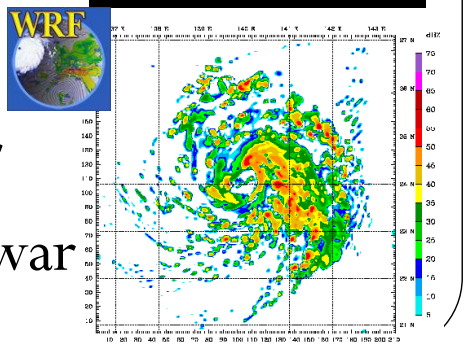


- Software is programs running on the hardware
  - Google Chrome (web browser)



  - Snapchat (application)

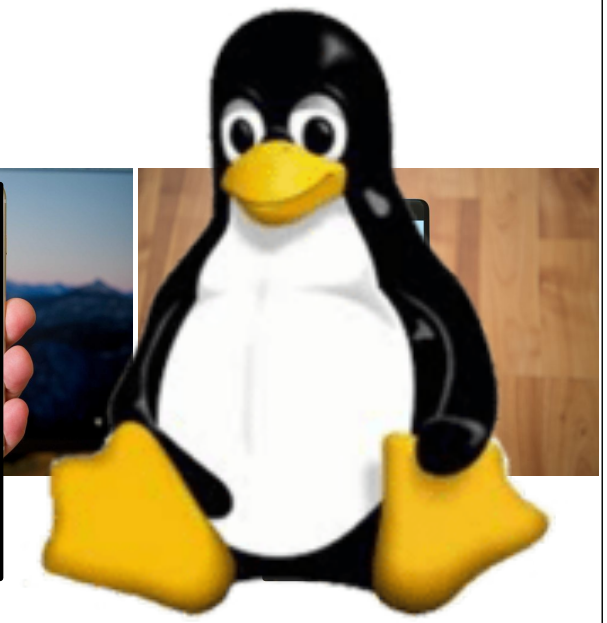
  - A WRF simulation of Typhoon Mawar



What is an operating system?

- The operating system is a piece of software that makes it easy for programs and other software to communicate with and make use of hardware
- Examples of operating systems:

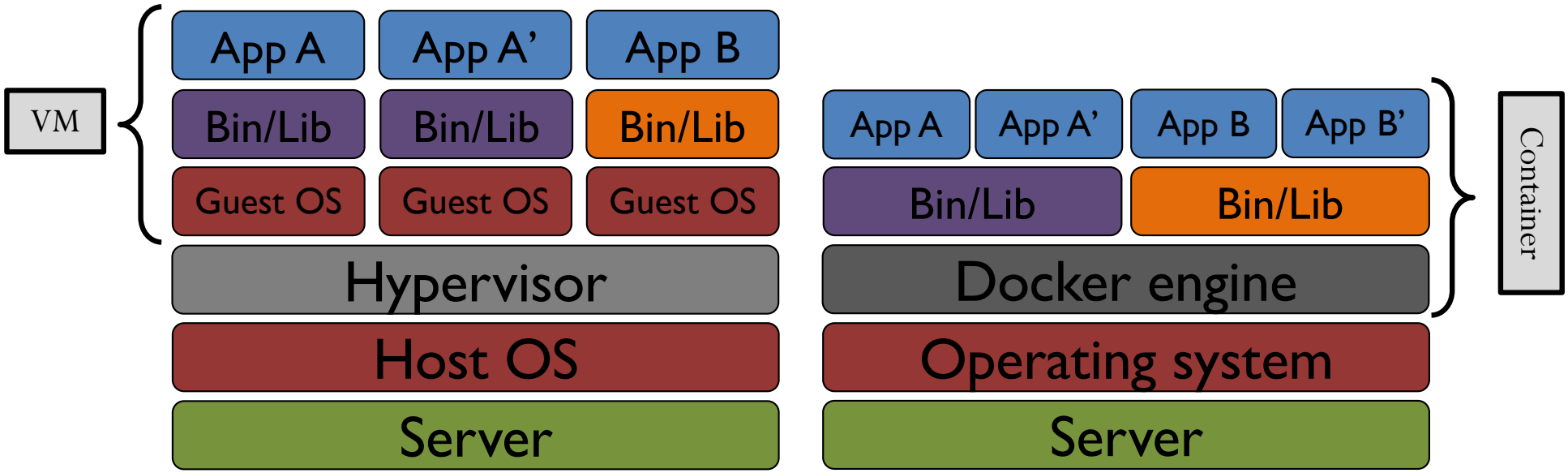
And of course, Linux



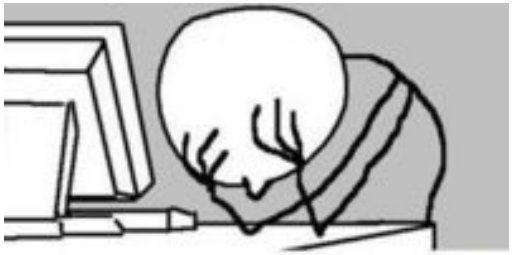
- Just like any other piece of software, an operating system can run another operating system: this is known as a Virtual Machine
  - Incredibly useful for a lot of applications, especially for software developers who have to work on multiple types of hardware and operating systems

What is a software container?

- A container is a self-contained "box" that allows you to essentially build software once and run it anywhere, so long as you can run the software that runs that "box"
- Virtual machines are great, but they are not ideal as they consume a lot of resources, and can have many portability problems on vastly different machines
- Containers are similar to a virtual machine, but much more lightweight and portable

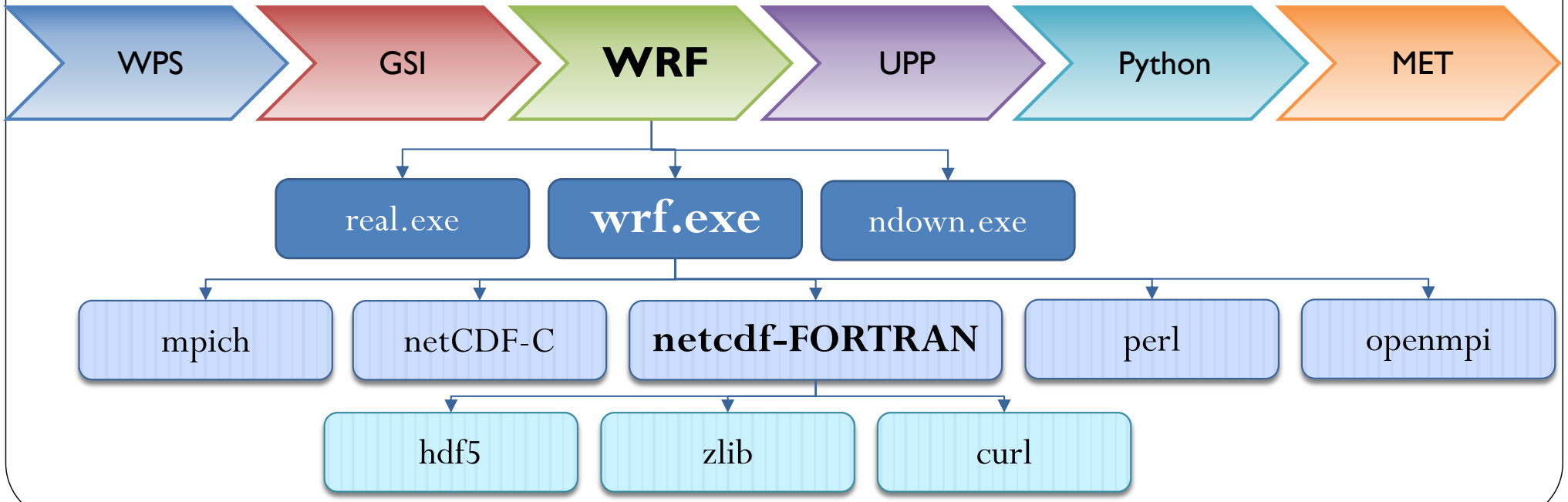


Why use containers?

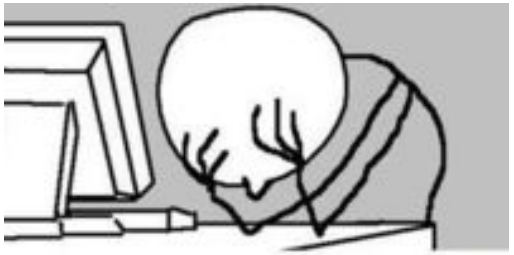


Stick figure trying to compile WRF, c. 2017

- Numerical weather prediction systems are *really* complicated
  - Many different components
  - Most components have multiple programs
  - Each of those programs depend on many *other* programs or software libraries
  - Compiling and setting up any one of these components has a chance to go horribly wrong

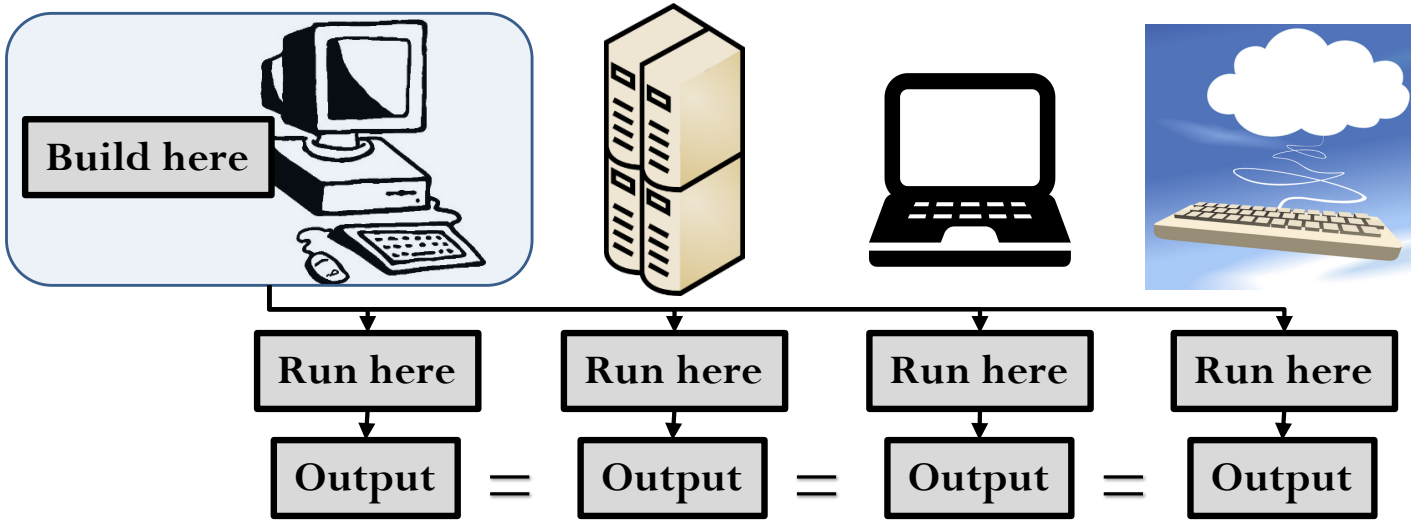


Why use containers?



Stick figure trying to compile WRF, c. 2017

- Containers mean someone still has to do all the work to get all those things set up… but only once!
  - Everything required for NWP can be packaged into isolated components, ready for development, shipment, and deployment to many different computing environments
  - Software *should* always run the same, regardless of where it is deployed

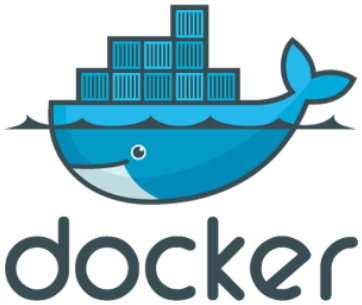


But I thought we just did this…

- If we have this AMI in the cloud that is completely set up with everything we need whenever we need it, why do we need to also put all these things in a container?
  - The cloud costs money to use it; computers you already have are free
  - Docker containers allow you to move your software environment to completely different machines, not just keeping it on a cloud instance
  - You can do scripting, debugging, visualization, and other low-resource tasks on your laptop for free, and then when it's time to call in the big guns, move it up to the cloud
  - It also allows you to switch cloud providers if you get a better deal elsewhere

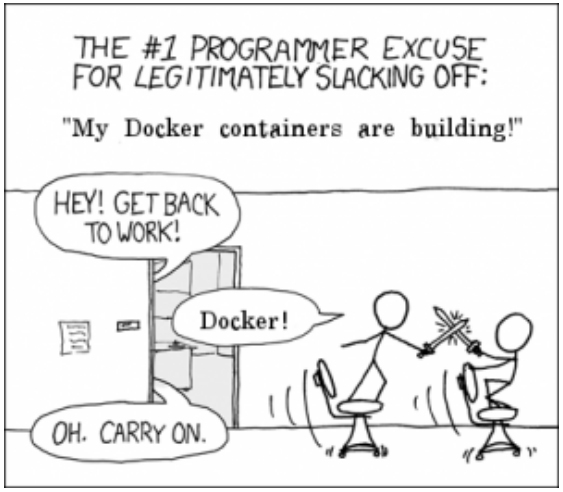
What is Docker?

- Docker is one of the leading software containerization platforms
  - Home page: **https://www.docker.com**
  - Documentation: **https://docs.docker.com**
- Works on Windows, Mac, and Linux machines



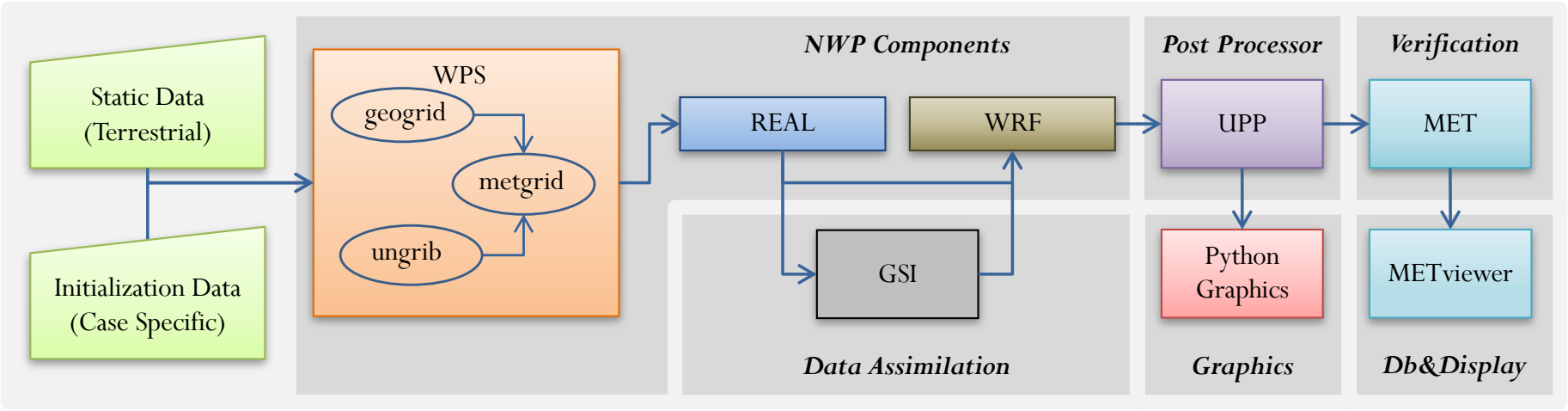
Understanding the lingo: Images & containers

- Image:
  - Inert, immutable snapshot
  - Created with the `docker build` command
  - Can build from scratch *(more setup, but offers customization!)* or save to a tar file, which can then be loaded for faster deployment
  - Will produce a container when started with run command
- Container:
  - Instance of an image created with the `docker run` command
  - Can be manipulated just like any other operating system, and data can be saved outside of the container with proper settings
  - Can have many running containers of the same image



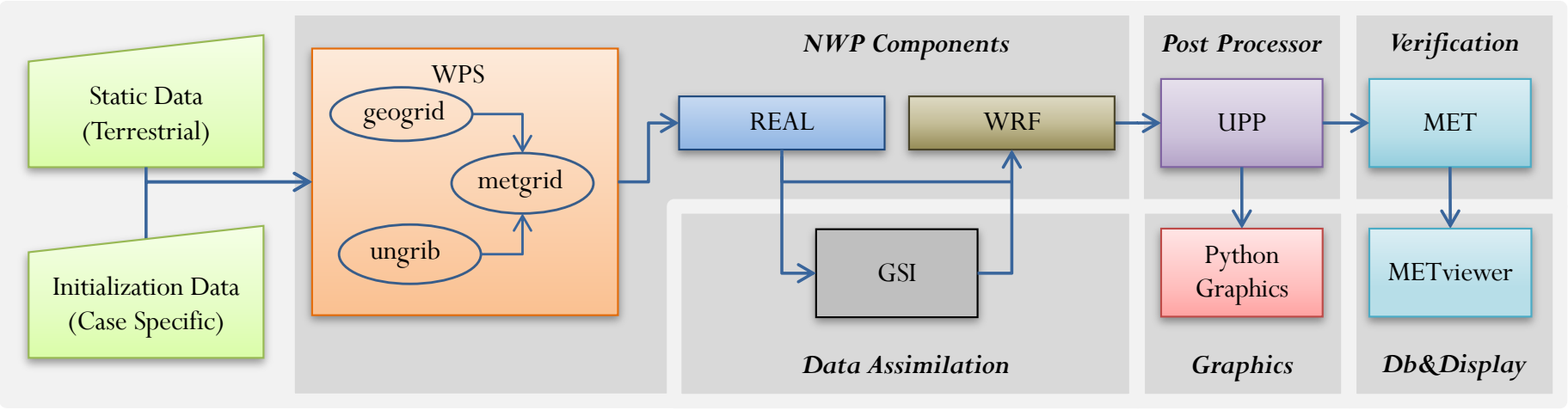
"The image is the recipe, the container is the cake"
- some rando on the internet

Introduction to end-to-end NWP components

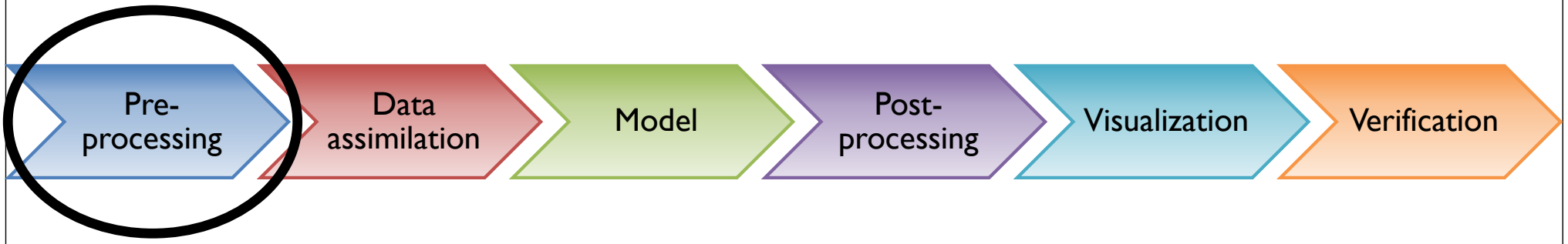


What is in the DTC containers?

- DTC containers package everything that is needed to initialize and run the WRF model, and produce graphics and verification from the WRF model output
  - Repository: https://github.com/NCAR/container-dtc-nwp/
  - Components can be run individually or as part of an entire workflow
  - Uses open source software such as GNU compilers; most components can be run in parallel
- README files and online tutorial with explicit instructions for building and running
- Necessary namelist and configuration files
- Data for sample cases provided (model IC/BCs, observation data for DA and verification)

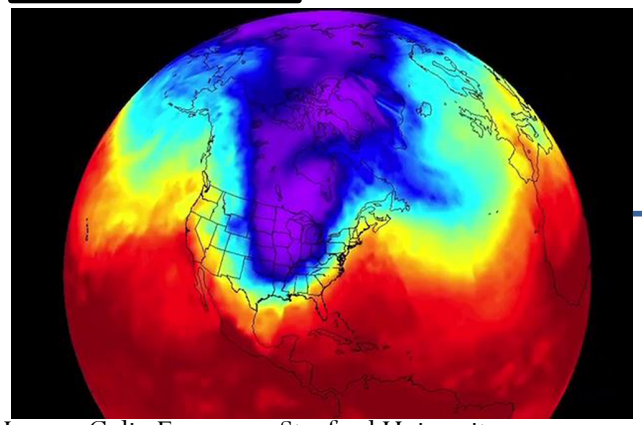


WRF Preprocessing System (WPS)



- The WRF Preprocessing System (WPS) takes existing 4-d atmospheric data from GRIB-format files and interpolates it onto the user's specified WRF domain grid.
  - Initial conditions: 3-dimensional wind, temperature, geopotential height and RH, 2-dimensional surface pressure
    - Typically a Global NWP forecast or reanalysis
  - Boundary conditions for the parent domain for the full length of the forecast

Global input



WRF limited-area domain

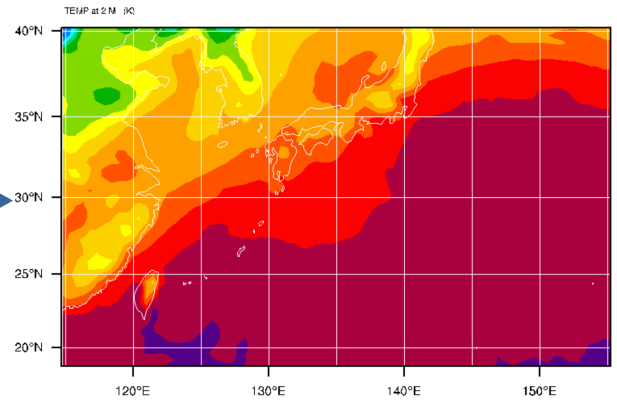
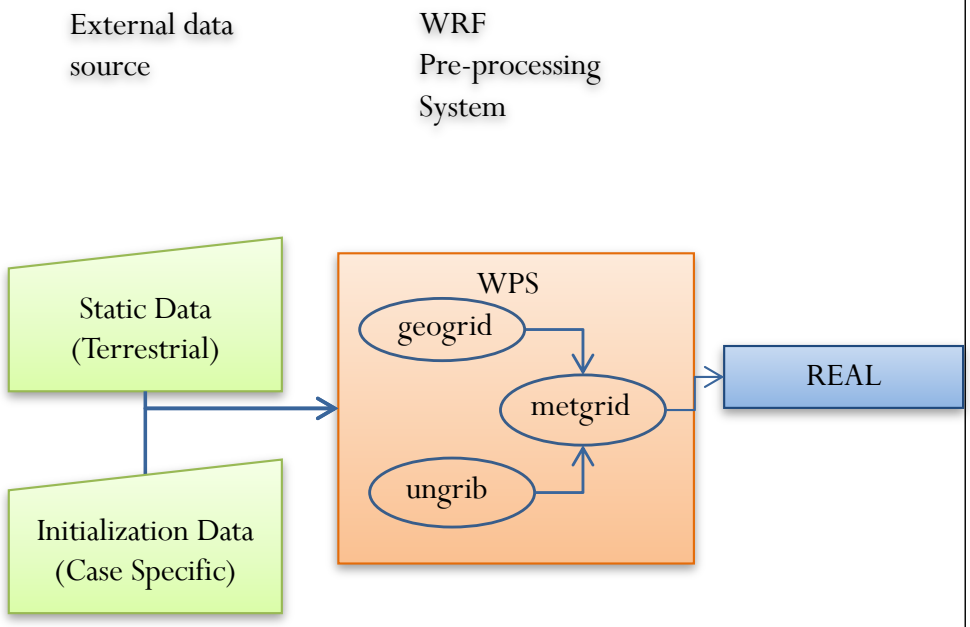


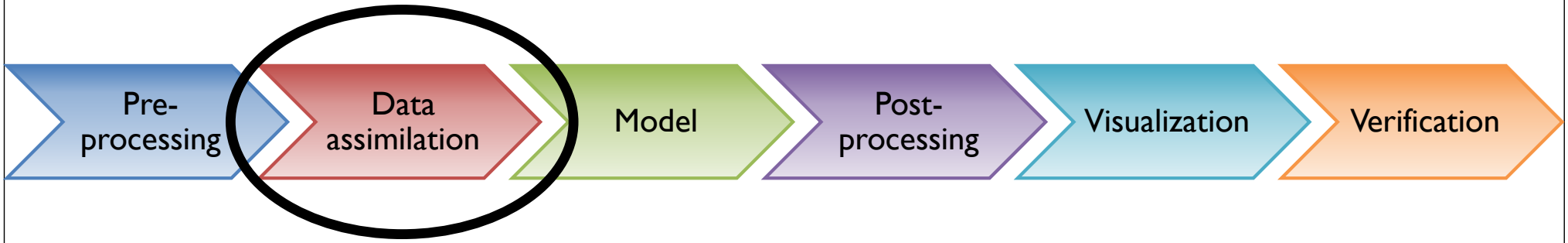
Image: Colin Epperson, Stanford University

Function of WPS components

- geogrid.exe (think **geo**graphical)
  - Define size/location of coarse domain and interpolate static terrestrial fields to coarse-domain and nested-domain grids
- ungrib.exe
  - Extract meteorological fields from GRIB files
- metgrid.exe (think **met**eorological)
  - Horizontally interpolate meteorological fields (from ungrib) to coarse grid (defined by geogrid)



Gridpoint Statistical Interpolation (GSI) data assimilation



- Data assimilation is the process by which real observations are incorporated into the initial conditions of your model to produce a better guess of the atmosphere's initial state
- For WRF applications, GSI can use the output from real.exe (i.e., wrfinput_d01) as the background field, and update it using the various observations. The updated background field – so called GSI analysis, can then be used as the initial conditions for WRF forecasts
- GSI can also use the WRF forecast files (i.e., wrfout_d01_<yyyymm-dd_hh:mm:ss>) as the background fields and update it for further forecasts.
- The observations can include conventional observations, satellite radiance, GPS radio occultations, etc.

What does GSI do?

GSI background (i.e.,wrfinput_d01)

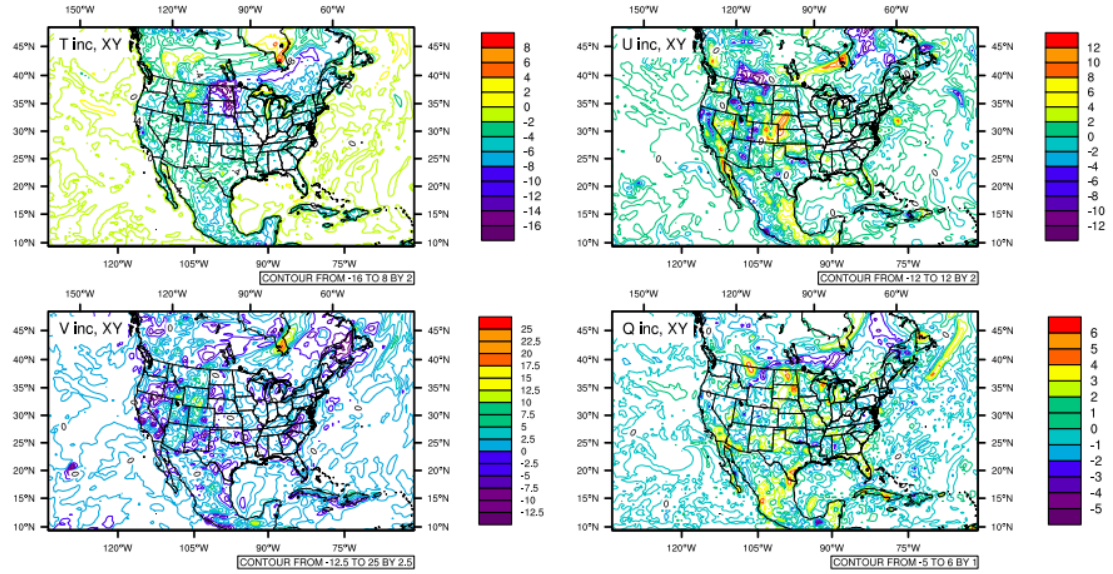
Observations in PrepBUFR/BUFR format (i.e., rap.t12z.prepbufr.tm00

Fix files (background errors, observation errors, CRTM coefficients, data selection, quality control, etc)

GSI executable (gsi.x)

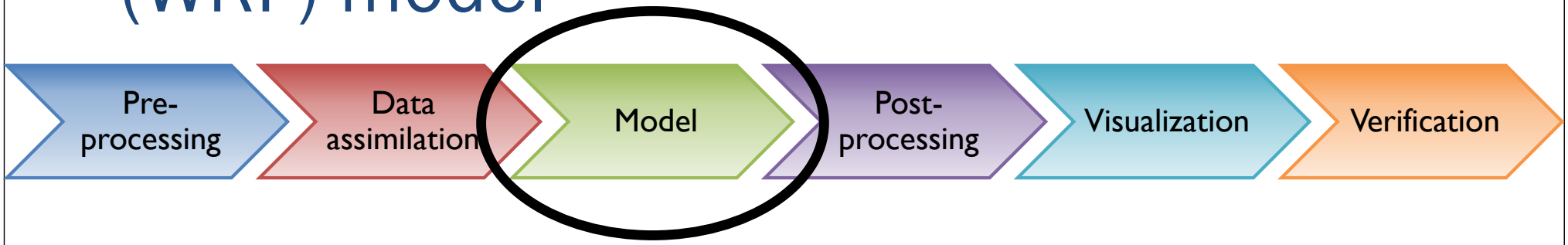
GSI analysis

WRF

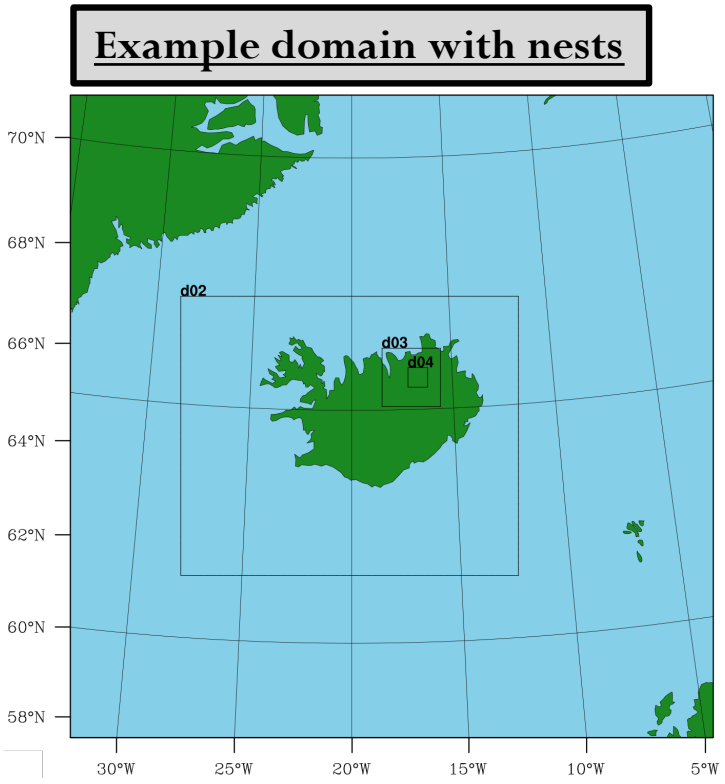


GSI analysis increment (analysis-background) after assimilating conventional observations

Weather Research and Forecasting (WRF) model

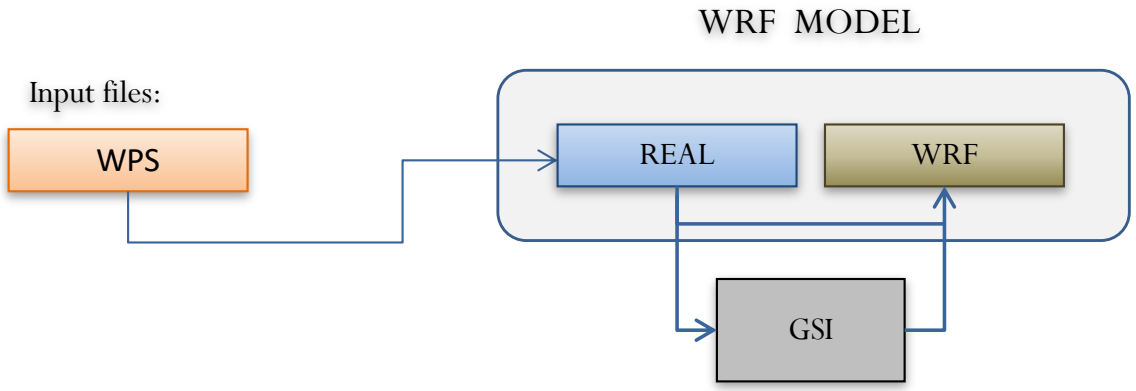


- Highly configurable, but also caters to less advanced users
  - e.g. 26 different microphysics schemes, 10 surface layer schemes, etc.
  - "Suites" of widely-used and tested scheme combinations are provided for casual users
  - Most options can be easily changed at runtime (no re-compilation required)
- Typically run for regional domains
  - One parent domain gets its initial and boundary conditions from the WRF Preprocessing System (WPS)
  - Can also have one or more child domains that get their boundary conditions from the parent domain

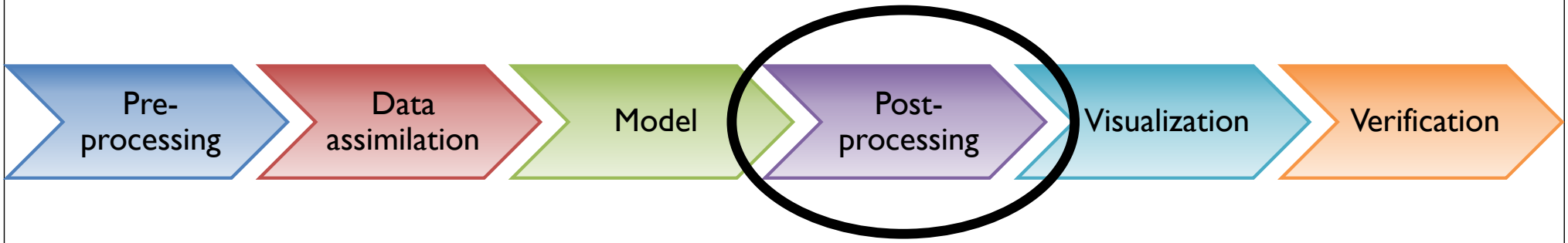


Function of WRF components

- real.exe
  - Generation of initial state for each of the requested domains
  - Creation of a lateral boundary file for the most coarse domain
  - Vertical interpolation for 3d meteorological fields and for sub-surface soil data
- wrf.exe
  - Forecast model integration through time



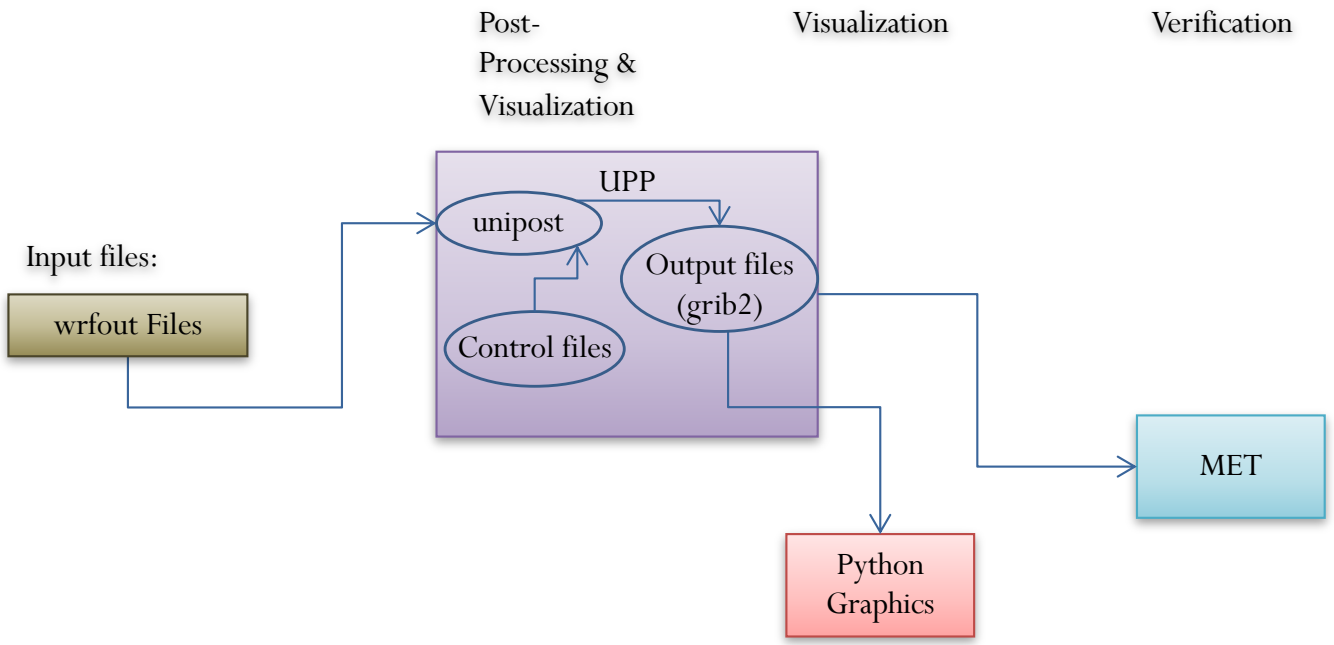
Unified Post Processor (UPP)



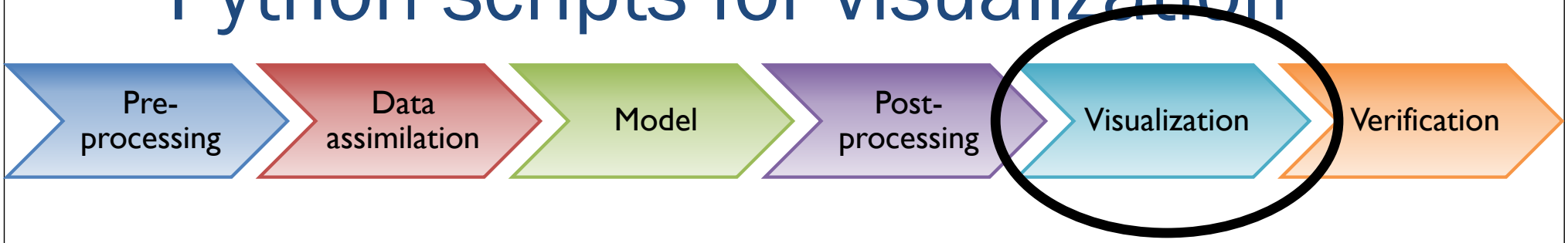
- The Unified Post-Processor (UPP) is a post-processor for WRF and other models
  - Developed at the National Centers for Environmental Prediction (NCEP) for use in its operational forecasting
  - Also available for community use and development with WRF
- Processes raw model output to more useful forms
  - Produces hundreds of products like those used operationally
    - T, Z, humidity, wind, cloud water, cloud ice, rain, and snow on isobaric levels
    - SLP + shelter level T, humidity, and wind fields
    - Precipitation-related fields
    - PBL-related fields
    - Diagnostic products (i.e. RH, radar reflectivity, CAPE)
    - Radiative/Surface fluxes 7) Cloud related fields 8) Aviation products
    - Synthetic satellite products
  - Creates output that can be plotted with your favorite visualization tool

Function of UPP

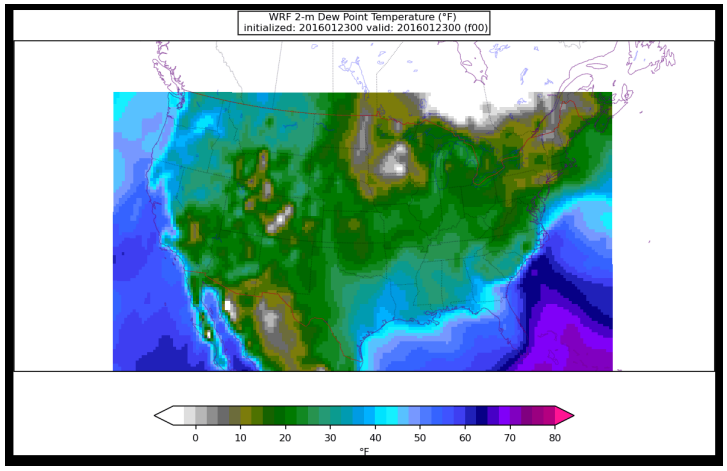
- unipost.exe
  - Performs vertical interpolation from model levels/surfaces onto isobaric, height, and other levels/surfaces
  - Calculated derived quantities/diagnostic fields
  - De-staggers wind onto mass points



Python scripts for visualization

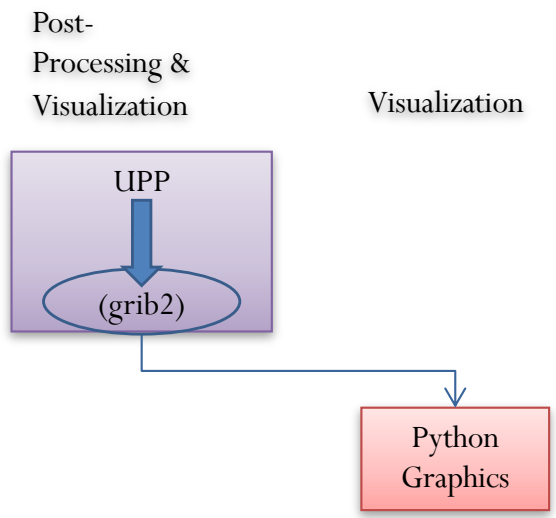


- Python is an open source programming language that was made to be easy-to-read and powerful for mathematical computation, graphics, and many other uses
- For our purposes, python is used for generating graphics from the UPP-processed output in basic image formats (.png and .gif)
- These images can be displayed using your favorite image viewing software
  - Depending on your machine and connection, it may be necessary to copy these images to your local machine for viewing

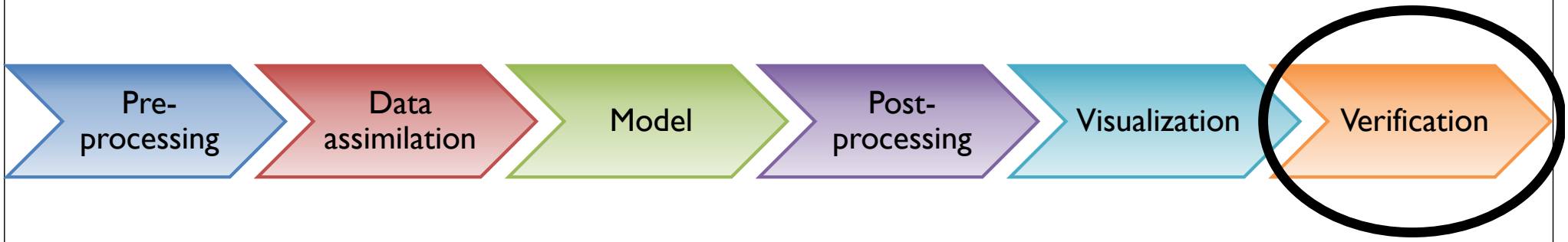


Function of Python component

- Python scripts
  - Read grib files created by UPP
    - Create plots of the variables of interest for each timestep
- A wrapper script stitches static images into .gif animations



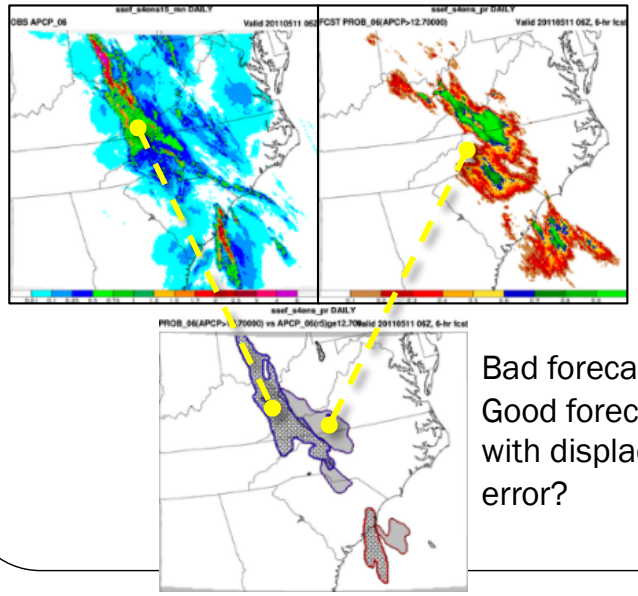
Model Evaluation Tools (MET) verification and visualization (METviewer)



What is MET?

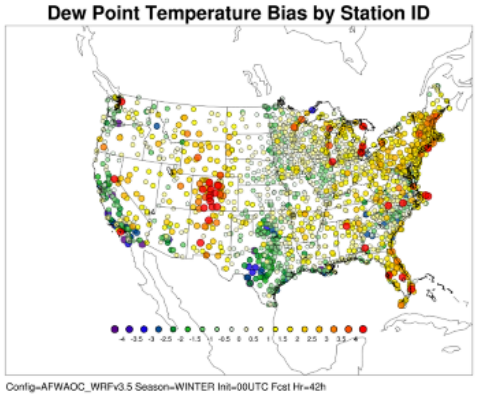
- **MET** is the statistics generation component of the larger METplus vx system
  - Freely available and supported by the DTC
  - Computes over 85 statistics using point, gridded, and tropical cyclone datasets
  - Full suite of standard statistics with cutting-edge statistics regularly added
  - Supports feature-based, ensemble, and tropical cyclone verification
  - Highlights include multiple interpolation methods, automated regridding, complex masking, and python embedding

Object Based and Spatial Methods

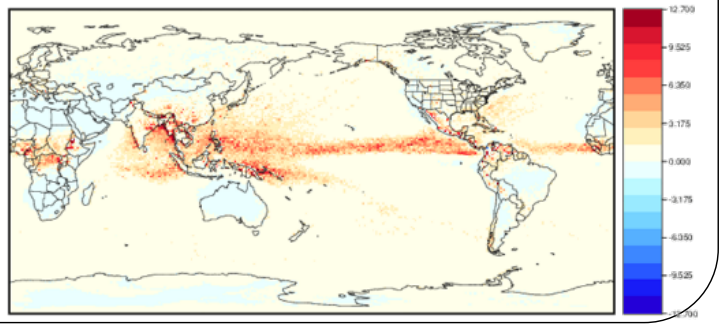


Bad forecast or Good forecast with displacement error?

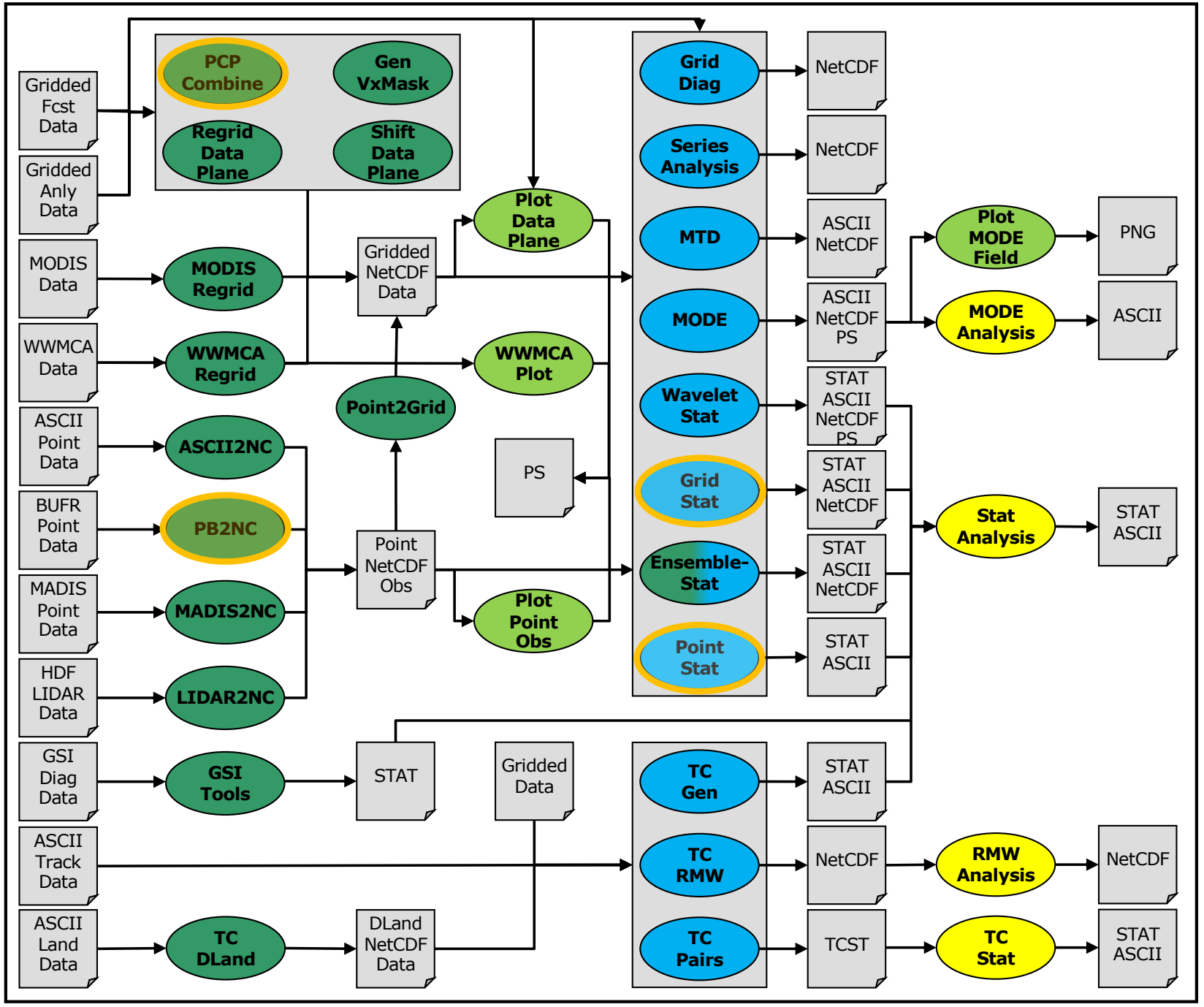
Geographical Representation of Errors



90th Percentile of difference between two models

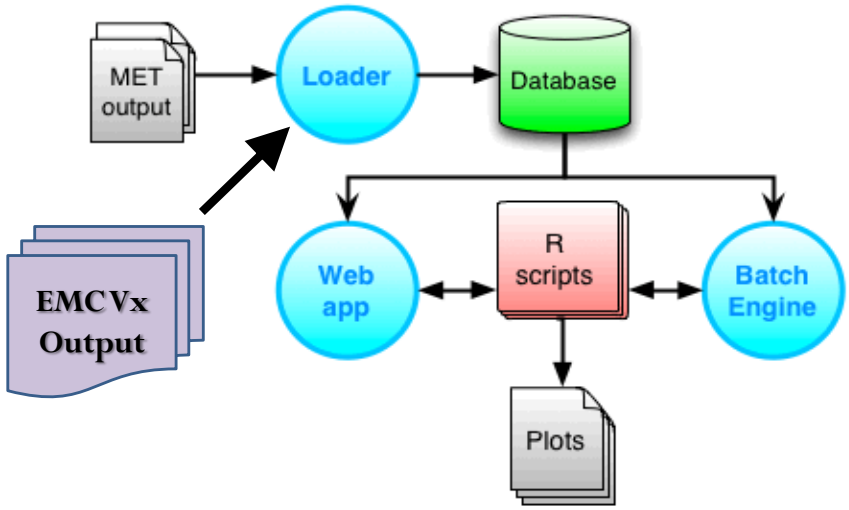


MET Overview v9.0

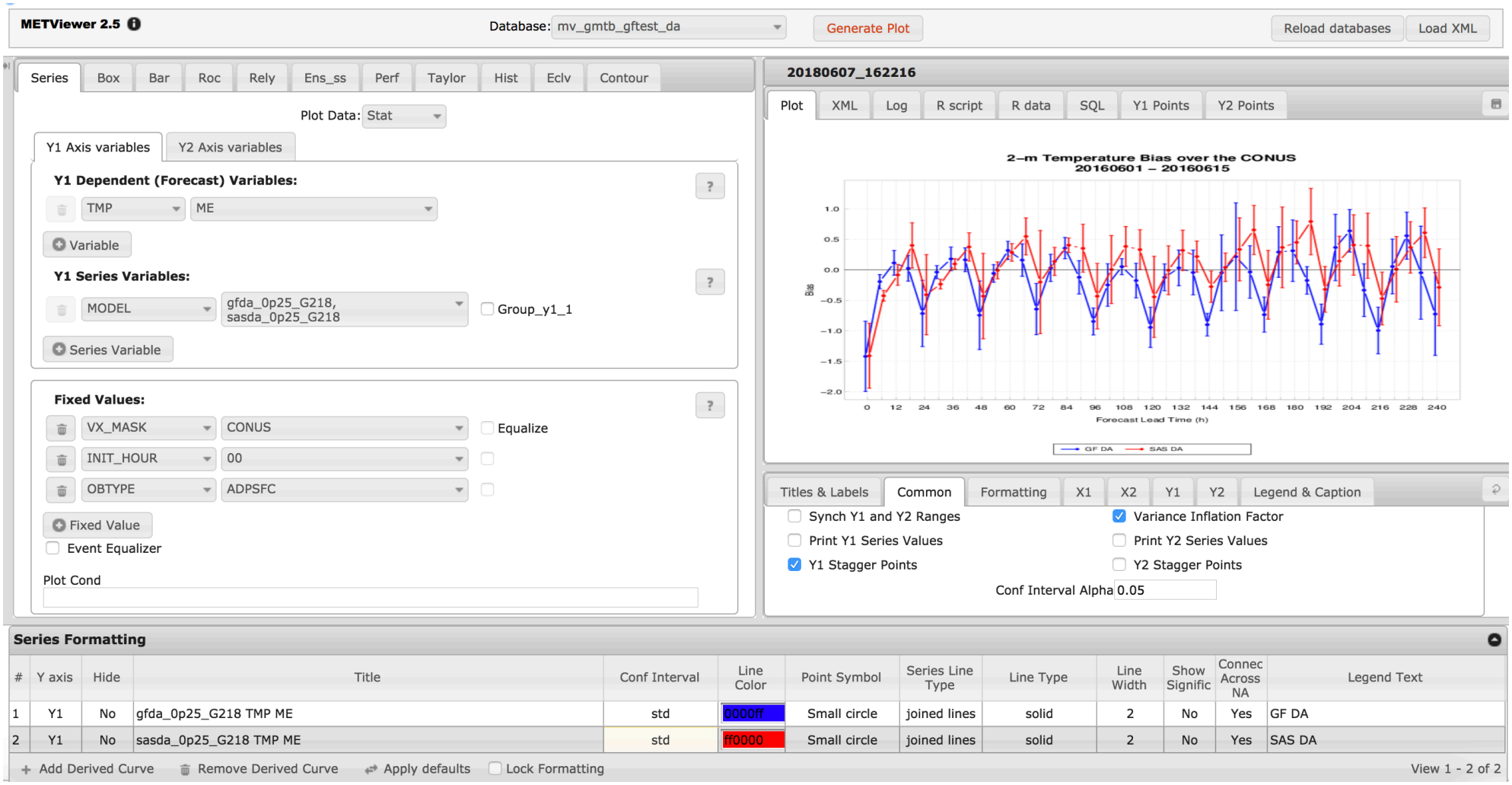


What is METviewer?

- **METviewer** is the database and display component of the larger METplus vx system
  - Freely available and supported by the DTC
  - Aggregates statistics across multiple runs and plots results via a batch system or interactive web GUI
  - Supports multiple plot types with new ones regularly added
  - Highlights include event equalization, pairwise differencing, statistical significance, and scorecard generation
  - Built with Java, Apache/Tomcat, MySQL, R statistics (transitioning to Python)



METviewer Interactive GUI



Software Packages Links

- WPS and WRF
  - Users' Page: **http://www2.mmm.ucar.edu/wrf/users/**
  - Online Tutorial: **http://www2.mmm.ucar.edu/wrf/OnLineTutorial/index.htm**
- GSI
  - Users' Page: **https://dtcenter.org/com-GSI/users/**
  - Online Tutorial: **https://dtcenter.org/com-GSI/users/tutorial/index.php**
- UPP
  - Users' Page: **https://dtcenter.org/community-code/unified-post-processor-upp**
  - Online Tutorial: **https://dtcenter.org/community-code/unified-post-processor-upp/upp-online-tutorial-uppv4-0**
- Python scripts
  - Not used in this workflow, but WRF-python is a very useful tool for manipulating WRF output specifically: https://wrf-python.readthedocs.io/en/latest/
- MET
  - Users' Page: **https://dtcenter.org/community-code/model-evaluation-tools-met**
  - Online Tutorial: **https://dtcenter.org/community-code/model-evaluation-tools-met/online-tutorial**

Overview of technologies

- ✓ Introduction and motivation
- ✓ Cloud computing (Amazon Web Services)
- ✓ Containerization software (Docker container)
- ✓ NWP workflow and components

Questions?