NASA Unified Weather Research and Forecasting Model (NU-WRF)
A Tutorial
For Bjerknes Patch 4 Release

NUWRF software integration group

June 28, 2017
Introduction
This tutorial describes the process of downloading, compiling, and running NU-WRF (Bjerknes patch 4 release and later) for four different and representative workflows:

- **Basic**: Based on WRF ARW.
- **Default**: The recommended non-chemistry NU-WRF workflow.
- **Chemistry**: The recommended chemistry (non-LIS) NU-WRF workflow.
- **SCM**: The Single Column Model NU-WRF workflow.

Each workflow consists of a series of steps starting with the acquisition of data files followed by the execution of several NU-WRF components. Successful completion of any workflow requires that **all** steps be completed in the order presented.

For any additional information please consult the [NU-WRF user’s guide](https://nuwrf.gsfc.nasa.gov/sites/default/files/docs/nuwrf_userguide.pdf) at:
Special instructions

- All workflows are designed with pre-defined settings\(^1\), including datasets. So, please refrain from editing the inputs unless you know what you are doing.

- It is assumed that users are running on Discover or Pleiades\(^2\), using the bash shell. However, the steps described here should work under other shells.

**IMPORTANT**: All lines starting with ‘\(>\)’ are Unix commands to be executed by the users.

\(^1\)That is, compiler/MPI combination, external libraries and run-time settings.

\(^2\)Only the basic workflow is available on Pleiades.
Useful links

- **Community ARW User Guide**
  [http://www2.mmm.ucar.edu/wrf/users/docs/user_guide_V3.5/contents.html](http://www2.mmm.ucar.edu/wrf/users/docs/user_guide_V3.5/contents.html)

- **Community WRF ARW Online Tutorial**

- **Documentation for using Discover and other NCCS systems**
  [http://www.nccs.nasa.gov/primer/](http://www.nccs.nasa.gov/primer/)

- **Documentation for using Pleiades and other NAS systems**

**NOTE:** The details for building the software on Discover and Pleiades are automatically handled by NUWRF’s build scripts!
Before starting any workflow there are a few steps that must be completed:

- Download or checkout the NU-WRF code.
- Build the NU-WRF executable components.
- Set some environment variables.
Download the code

Tar files are available to NCCS and NAS s0942 group members.
On DISCOVER:

> cp /discover/nobackup/projects/nu-wrf/releases/stable/<tarball> /some/path

where tarball is one of the following:
nu-wrf_v8p4-wrf371-lis71rp7.tgz
nu-wrf_v8p4-wrf371-lis71rp7.tar.bz2

On PLEIADES:

> cp /nobackupp8/nuwrf/releases/stable/<tarball> /some/path

where tarball is one of the following:
nu-wrf_v8p4-wrf371-lis71rp7.tar.bz2
nu-wrf_v8p4-wrf371-lis71rp7.tgz

To untar type:

> tar -zxf nu-wrf_v8p4-wrf371-lis71rp7.tgz
or
> bunzip2 nu-wrf_v8p4-wrf371-lis71rp7.tar.bz2
> tar -xf nu-wrf_v8p4-wrf371-lis71rp7.tar


**Developers** can check out the code from the SVN repository:

> `svn checkout svn+ssh://progressdirect/svn/nu-wrf/code/tags/releases/v8p4-wrf371-lis71rp7`

or

> `svn checkout svn+ssh://progress.nccs.nasa.gov/svn/nu-wrf/code/tags/releases/v8p4-wrf371-lis71rp7`

Or, the development branch, from the Git repository:

> `git clone progressdirect:/git/projects/nu-wrf`

For more information refer to the **NU-WRF user’s guide**:

Build NU-WRF

First log in to a Discover SP3 node or Pleiades front end node. Create environment variable NUWRFDIR that defines directory path of NU-WRF code downloaded earlier. Then, cd to it. For example, on Discover:

> export NUWRFDIR=/discover/nobackup/user_id/nu-wrf
> cd $NUWRFDIR

Now execute the build script as follows (runs in background, saves output in log.out):

**Basic workflow:**

> ./build.sh wrf wps >& log.out &

**Default workflow:**

> ./build.sh wrf wps lis ldt lisconfig >& log.out &

**Chemistry workflow:**

> ./build.sh chem wps utils >& log.out &

**SCM workflow:**

> ./build.sh ideal_scm_lis_xy wps lis ldt lis4scm >& log.out &
Build NU-WRF

After compilation - assuming there are no errors - several executables will be created. For example:

WRF executables:

$NUWRFDIR/WRFV3/main/real.exe
$NUWRFDIR/WRFV3/main/wrf.exe

WPS executables:

$NUWRFDIR/WPS/geogrid.exe
$NUWRFDIR/WPS/metgrid.exe
$NUWRFDIR/WPS/ungrib.exe

For example, to see the names of all the executables created with an exe extension run:

> find $NUWRFDIR -name \\*.exe
Environment variables

We already set **NUWRFDIR**, but we need two more variables.

- Select a directory for running the model and set it equal to the **RUNDIR** environment variable. For example:

  ```
  > export RUNDIR=/discover/nobackup/user_id/scratch/basic_workflow
  ```

  Make sure you create **RUNDIR** outside of **NUWRFDIR**. This is useful when switching between NU-WRF versions or for updating to new changes.

- Finally, set the **PROJECTDIR** environment variable:

  ```
  > export PROJECTDIR=/discover/nobackup/projects/nu-wrf
  ```

Please note that these variables (**NUWRFDIR**, **RUNDIR**, **PROJECTDIR**) are used in all workflows.
Basic Workflow
A NU-WRF workflow consists of running various pre-processing (WPS) components as well as the main NU-WRF model. The basic workflow described next is the simplest approach to running simulations with NU-WRF. It consists of running the WRF Pre-processing System (WPS) plus an additional WRF pre-processor (REAL) before running the WRF model.

*Neither chemistry nor advanced land surface initialization are used.*
NU-WRF basic workflow (similar to WRF ARW).

WPS
WRF Preprocessing System

WRF-ARW
(Advanced Research WRF)

GEOGRID → METGRID → REAL → WRF

UNGGRID
Required files for basic workflow

Copy the following files to **RUNDIR**:

```
> cp -r $PROJECTDIR/tutorial/basic $RUNDIR
```

Where:
- `common.cfg` : shared script with settings used by other scripts.
- `run*.bash` : scripts to run pre-processors and model.
- `namelist*` : namelist files required by executables.
- `data/ungrib/*` : GRIB atmospheric data for initial conditions used by UNGRIB component.
Required script changes

> cd $RUNDIR

Use your favorite editor to edit `common.cfg` and change the values of NUWRFDIR and RUNDIR using the values set earlier.

```bash
# *** Please make sure these settings are correct ***
# NUWRFDIR specifies the location of the NU-WRF source code
NUWRFDIR=<CHANGE THIS>
# RUNDIR specifies the location of the temporary run directory
RUNDIR=<CHANGE THIS>
```

You may need to edit all the run* files’ account information and other settings. However, if you belong to group s0492 then the scripts should work without any modifications.

Change account to appropriate SBU charge code:
```
#SBATCH --account s0942
```
Change if you want to change number of nodes, hasw - to run on haswell nodes:
```
#SBATCH --ntasks=16 --constraint=hasw
```
Uncomment and set according to your needs and privileges:
```
##SBATCH --qos=high
```
Uncomment (if desired) and substitute your e-mail here:
```
##SBATCH --mail-user=user@nasa.gov
```
A note about namelists settings

Things to keep in mind before we run NU-WRF components.

- The length of the simulations is specified in the namelist files:
  - In namelist.wps the length is determined by start_date and end_date
  - In namelist.input look for start_ and end_ fields.
  - The dates in both namelists must be consistent.

- The workflow is designed to work as-is. However, if you want to run for different dates:
  - You must get the corresponding atmospheric data for initial conditions.
  - Yo may need to modify the namelists. For example in namelist.input, make sure end_day - start_day = run_days.

- For **any** other changes please refer to the user’s guide.
GEOGRID interpolates static and climatological terrestrial data (land use, albedo, vegetation greenness, etc) to each WRF grid.

- Input: namelist.wps
- Output: For $N$ domains (max_dom in namelist.wps), $N$ geo_em files will be created.

To run:

> cd $RUNDIR
> sbatch run_geogrid.bash

---

When done, check for "Successful completion" string in the file geogrid.slurm.out. geogrid.log.<node> will also be created for tracking run failures or debugging.
UNGRIB reads GRIB or GRIB2 files with dynamic meteorological and dynamic terrestrial data (soil moisture, soil temperature, sea surface temperature, sea ice, etc) and writes specific fields in a WPS intermediate format.

- Input: namelist.wps and GRIB input data.
- Output: Several FNL* files corresponding to number of intervals (interval_seconds) in simulation length (start/end dates).

**NOTE:** The GRIB input is referenced in the run script, run_ungrib.bash:

```
./link_grib.csh data/ungrib/fnl_`
```

The UNGRIB output (FNL) is determined by the settings in the WPS namelist (namelist.wps).

To run:

```bash
> cd $RUNDIR
> ./run_ungrib.bash
or
> sbatch run_ungrib.bash
```

When done, check for "Successful completion" string in file ungrib_logs/ungrib.log.
METGRID horizontally interpolates the output from UNGRIB to the WRF domains, and combines it with the output from GEOGRID.

- Input: namelist.wps, UNGRIB output, geo_em* files.
- Output: Several met_em* files corresponding to number of intervals (interval_seconds) in simulation length (start/end dates).

To run:

> cd $RUNDIR
> sbatch run_metgrid.bash

When done, check for "Successful completion" string in the file metgrid.slurm.out. metgrid.log.<node> will also be created for tracking run failures or debugging.
REAL vertically interpolates the METGRID output to the WRF grid, and creates initial and lateral boundary condition files.

- **Input:** namelist.input, met_em* files, geo_em* files.
- **Output:** wrfinput* files (one for each domain), wrfbdy_d01.

To run:

```bash
> cd $RUNDIR
> sbatch run_real.bash
```

Check real.slurm.out for run completion.
Also check real.rsl.out.<node> and real.rsl.error.<node> for tracking run failures or debugging.
This program will perform a numerical weather prediction simulation using the data from REAL.

- **Input:** wrfinput* files (one for each domain), wrfbdy_d01.
- **Output:** wrfout* and wrfrst* files (one for each domain).

To run:

```
> cd $RUNDIR
> sbatch run_wrf.bash
```

Check wrf.slurm.out for run completion.
Also check wrf.rsl.out.<node> and wrf.rsl.error.<node> for tracking run failures or debugging.
Post-processing on Discover

Using NCVIEW:

WRF output files (NETCDF4) can be viewed using a special version of ncview installed on Discover:

/usr/local/other/SLES11/ncview/2.1.1/bin/ncview <filename>
Using RIP (NCAR graphics). Submit the `rip` job:

```bash
> cd $RUNDIR
> sbatch run_rip.bash # (or use sbatch)
> idt filename.cgm # Substitute actual filename
```

run_rip.bash will run ripdp_wrfarw and rip to generate NCAR Graphics cgm files.

`idt` is a NCAR Graphics executable in `$NCARG_ROOT/bin`

Sample RIP plot specification tables are in `$NUWRFDIR/scripts/rip`

and are looped through by run_rip.bash

See http://www2.mmm.ucar.edu/wrf/users/docs/ripug.htm for info on customizing plots with RIP.

Minor changes to run_rip.bash may be necessary.
Other available community software packages are ARWPOST (for GRADS), UPP (for GRIB), and MET (for atmospheric verification).

For more information on community post-processing packages available with WRF, see

http://www2.mmm.ucar.edu/wrf/users/docs/user_guide_V3.5/
users_guide_chap9.htm

ARW user homepage:

http://www2.mmm.ucar.edu/wrf/users

NUWRF specific post-processors are GSDSU (simulates satellite data) and LVT (land surface verification).
On Pleiades set PROJECTDIR to:

> export PROJECTDIR=/nobackupp8/nuwrf

Edit all the run* files for account information changes. If you belong to group s0492 then the scripts should work without any modifications.

Change s0942 to appropriate SBU charge code:

#PBS -W group_list=s0942

Change if you want to change number of nodes, "has" - to run on haswell nodes:

#PBS -l select=6:ncpus=16:mpiprocs=16:model=san

Uncomment and set according to your needs and privileges:

##PBS -q devel

Uncomment (if desired) and substitute your e-mail here:

##PBS -M=user@nasa.gov
Default Workflow
The default workflow is a more advanced approach to running simulations with NU-WRF. Instead of using land surface fields interpolated from a coarser model or reanalysis, a custom-made land surface state is created by LIS on the same grid and with the same terrestrial data and land surface physics as WRF. WRF then calls LIS on each advective time step, and provides atmospheric forcing data and receives land surface data (fluxes, albedo, etc) in return.

*No chemistry is used.*
NU-WRF default workflow.

MERRA2WRF, SST2WRF, LIS spin up & WRF-LIS coupled workflow

This NU-WRF default workflow uses MERRA/ MERRA2 reanalysis atmospheric initial and lateral boundary conditions, RSS SST (Remote Sensing Systems SST product), and LIS land surface model initial conditions, and runs WRF coupled with LIS.
Copy the following files to **RUNDIR**:

```
> cp -r $PROJECTDIR/tutorial/default $RUNDIR
```

Where:
- `common.cfg`: shared script with settings used by other scripts.
- `run*.bash`: scripts to run pre-processors and model.
- `namelist*`: namelist files required by executables.
- `data/LIS_restarts`: restarts used to run WRF with LIS.
Required script changes

```bash
> cd $RUNDIR

Use your favorite editor to edit `common.cfg` and change the values of NUWRFDIR and RUNDIR using the values set earlier.

```bash
# *** Please make sure these settings are correct ***
# NUWRFDIR specifies the location of the NU-WRF source code
NUWRFDIR=<CHANGE THIS>
# RUNDIR specifies the location of the temporary run directory
RUNDIR=<CHANGE THIS>
```

You may need to edit all the run* files’ account information and other settings. However, if you belong to group s0492 then the scripts should work without any modifications.

Change account to appropriate SBU charge code:
```bash
#SBATCH --account s0942
```
Change if you want to change number of nodes, hasw - to run on haswell nodes:
```bash
#SBATCH --ntasks=16 --constraint=hasw
```
Uncomment and set according to your needs and privileges:
```bash
##SBATCH --qos=high
```
Uncomment (if desired) and substitute your e-mail here:
```bash
##SBATCH --mail-user=user@nasa.gov
```
A note about namelists settings

Things to keep in mind before we run NU-WRF components.

- The length of the simulations is specified in the namelist files:
  - In namelist.wps the length is determined by start_date and end_date
  - In namelist.input look for start_ and end_ fields.
  - The dates in both namelists must be consistent.

- The workflow is designed to work as-is. However, if you want to run for different dates:
  - You must get the corresponding atmospheric data for initial conditions.
  - You may need to modify the namelists. For example in namelist.input, make sure end_day - start_day = run_days.

- For any other changes please refer to the user’s guide.
GEOGRID interpolates static and climatological terrestrial data (land use, albedo, vegetation greenness, etc) to each WRF grid.

- Input: namelist.wps
- Output: For $N$ domains (max_dom in namelist.wps), $N$ geo_em files will be created.

To run:

```bash
> cd $RUNDIR
> sbatch run_geogrid.bash
```

When done, check for "Successful completion" string in the file geogrid.slurm.out. geogrid.log.<node> will also be created for tracking run failures or debugging.
Another source of NU-WRF initial and lateral boundary conditions is NASA’s GEOS-5 global model. A number of dataset options exist from GEOS-5, including archived MERRA and MERRA-2 reanalyses. However, there are some issues to keep in mind.

First of all the GEOS-5 land surface data cannot be used to initialize WRF, due to fundamental differences between the GEOS-5 Catchment LSM and those in WRF. Thus, the GEOS-5 data in the current workflow also includes WRF-LIS. Second, GEOS-5 writes output in netCDF (and historically HDF4 and HDFEOS2) instead of GRIB. And furthermore, GEOS-5 often does not output all the variables expected by WPS.

To address these issues, special preprocessing software has been developed: MERRA2WRF. MERRA2WRF is a monolithic program customized to process the 6-hourly reanalyses from MERRA and MERRA-2. For 3-hourly MERRA-2 processing users must fall back to GEOS2WRF, which is not discussed here.
MERRA2WRF scripts process the 6-hourly reanalyses from MERRA and MERRA-2. The scripts download data from the NASA GES DISC web servers. There are two cases - **for this tutorial we are doing case 2 below:**

Case 1: MERRA reanalysis for WRF initial and lateral boundary conditions.

```bash
> cd $RUNDIR
> cp $NUWRFDIR/utils/geos2wrf_2/scripts/run_merra/Run_MERRA.csh .
> ./Run_MERRA.csh 20070119 20070120 . $NUWRFDIR
> cp $RUNDIR/data/merra2wrf/MERRA* .
```

Case 2: MERRA2 reanalysis for WRF initial and lateral boundary conditions.

```bash
> cd $RUNDIR
> cp $NUWRFDIR/utils/geos2wrf_2/scripts/run_merra/Run_MERRA2.csh .
> ./Run_MERRA2.csh 20070119 20070120 . $NUWRFDIR
> cp $RUNDIR/data/merra2wrf/MERRA* $RUNDIR
```

To run:

> cd $RUNDIR
> mkdir RUN_SST2WRF
> cd RUN_SST2WRF
> cp $NUWRFDIR/utils/sst2wrf/scripts/Run_SST.csh .

For a start date = 20070119 and end date = 20070120:

> ./Run_SST.csh 20070119 20070120 mw_ir . $NUWRFDIR

SSTRSS:* files will be created, and these files should be copied to the RUNDIR before running METGRID component.

> cp $RUNDIR/RUN_SST2WRF/sstdata/mw_ir/SSTRSS* $RUNDIR
METGRID horizontally interpolates MERRA and SSTRSS data to the WRF domains, and combine them with the output from GEOGRID.

- Input: namelist.wps, MERRA*, SSTRSS* and geo_em* files.
- Output: Several met_em* files corresponding to number of intervals (interval_seconds) in simulation length (start/end dates).

To run:

> cd $RUNDIR
> sbatch run_metgrid.bash

---

When done, check for "Successful completion" string in the file metgrid.slurm.out. metgrid.log.<node> will also be created for tracking run failures or debugging.
LISCONFIG (optional step)

LISCONFIG is an optional utility for customizing LDT and LIS ASCII input files so their domain(s) (grid size, resolution, and map projection) match that of WRF. It uses the output from GEOGRID to determine the reference latitude and longitude. For this tutorial we are using pre-defined lis.config and ldt.config files, and the following steps are only shown for future reference.

- Input: lis.config, ldt.config and geo_em* files.
- Output: lis.config.new, ldt.config.new

To run:

> cd $RUNDIR
> cp $NUWRFDIR/utils/lisconfig/scripts/lisWrfDomain.py .
> ln -s $NUWRFDIR/utils/bin/lisWrfDomain.x ./EXE
> lisWrfDomain.py EXE lis.config ldt.config ./
LDT (pre-LIS)

LDT processes data inputs for different surface models. In this use-case we are using the Noah v3.6 land surface model.

- **Input**: ldt.config (ldt.config.prelis gets copied into ldt.config by run_ldt_prelis.bash)
- **Output**: lis_input* files for each NU-WRF domain.

To run:

```
> cd $RUNDIR
> sbatch run_ldt_prelis.bash
```

---

When done, check for "Finished LDT run" string in the file ldt_log_prelis.0000
LIS can be run in various modes. Here we run in "retrospective" mode and provide restart files as input to LIS (in data/LIS_restarts).

- **Input:** lis.config, LIS_RST_NOAH36*nc, forcing_variables_wrfcplmode.txt, NOAH36_OUTPUT_LIST_SPINUP.TBL
- **Output:** surface model output in OUTPUT directory.

To run:

```
> cd $RUNDIR
> sbatch run_lis.bash
---
```

Check this file for successful run completion: lis.slurm.out lislog.<node> will also be created for tracking run failures or debugging.

For more information about LIS see

https://modelingguru.nasa.gov/community/atmospheric/lis.
After running LIS, it is necessary to rerun LDT in “NUWRF preprocessing for real” mode. This requires modifications to ldt.config to specify the static output file from LDT and the dynamic output file from LIS. Fields from both will be combined and written to a new netCDF output file for use by REAL.

- Input: ldt.config (ldt.config.postlis gets copied into ldt.config by run_ldt_postlis.bash), LIS history files in OUTPUT directory.
- Output: lis4real_input* files for each NU-WRF domain.

To run:

> cd $RUNDIR
> sbatch run_ldt_prelis.bash

---

When done, check for "Finished LDT run" string in the file ldt_log_postlis.0000
REAL vertically interpolates the METGRID output to the WRF grid, and creates initial and lateral boundary condition files.

- Input: namelist.input, met_em* files, geo_em*, lis4real_input* files.
- Output: wrfinput* files (one for each domain), wrfbdy_d01.

To run:

\[ \text{> cd \$RUNDIR} \]
\[ \text{> sbatch run_real.bash} \]

---

Check real.slurm.out for run completion.
Also check real.rsl.out.<node> and real.rsl.error.<node> for tracking run failures or debugging.
Running WRF in this case is similar to the basic case, except that WRF will also read the lis.config file and the LIS restart files that were produced during the "retrospective" run.

- **Input**: wrfinput* files (one for each domain), wrfbdy_d01.
- **Output**: wrfout* and wrfrst* files (one for each domain).

---

To run:

> cd $RUNDIR
> cp namelist.input.wrf namelist.input
> cp lis.config.wrflis lis.config
> sbatch run_wrf.bash

Check wrf.slurm.out for run completion.

Also check wrf.rsl.out.<node> and wrf.rsl.error.<node> for tracking run failures/ debugging.
Using NCVIEW:

WRF output files (NETCDF4) can be viewed using a special version of ncview installed on Discover:

/usr/local/other/SLES11/ncview/2.1.1/bin/ncview <filename>
Using RIP (NCAR graphics). Submit the `rip` job:

```
> cd $RUNDIR
> sbatch run_rip.bash # (or use sbatch)
> idt filename.cgm # Substitute actual filename
```

run_rip.bash will run ripdp_wrfarw and rip to generate NCAR Graphics cgm files.

`idt` is a NCAR Graphics executable in `$NCARG_ROOT/bin`

Sample RIP plot specification tables are in `$NUWRFDIR/scripts/rip`
and are looped through by run_rip.bash

See [http://www2.mmm.ucar.edu/wrf/users/docs/ripug.htm](http://www2.mmm.ucar.edu/wrf/users/docs/ripug.htm) for info on customizing plots with RIP.

Minor changes to run_rip.bash may be necessary.
NU-WRF offers advanced aerosol modeling using the implementation of GOCART. This workflow is similar to the default workflow but excludes the LIS steps and instead adds the GOCART2WRF preprocessor for providing chemical boundary conditions from GEOS-5 as well as the community PREP_CHEM_SOURCES program for emissions.

GOCART2WRF supports four possible sources of GOCART data:
- offline GOCART
- on-line GOCART from GEOS5
- MERRAero reanalyses
- MERRA-2

We will describe the "offline GOCART" case only.

LIS is not used.
This NU-WRF CHEM workflow uses atmospheric initial and lateral boundary conditions from GRIB files, RSS SST (Remote Sensing Systems SST product), and chemical tracers using Prep-chem sources, Convert_emiss, aerosol data from GOCART or MERRA-Aero, and runs WRF-CHEM.
Copy the following files to **RUNDIR**:

```
cp -r $PROJECTDIR/tutorial/chemistry $RUNDIR
```

Where:
- `common.cfg` : shared script with settings used by other scripts.
- `run*.bash` : scripts to run pre-processors, model and post-processor.
- `namelist*` : namelist files required by executables.
- `data/ungrib` : GRIB atmospheric data for initial conditions used by UNGRIB component.
- `data/gocart` : data files used by GOCART
Required script changes

> cd $RUNDIR

Use your favorite editor to edit `common.cfg` and change the values of NUWRFDIR and RUNDIR using the values set earlier.

# *** Please make sure these settings are correct ***
# NUWRFDIR specifies the location of the NU-WRF source code
NUWRFDIR=<CHANGE THIS>
# RUNDIR specifies the location of the temporary run directory
RUNDIR=<CHANGE THIS>

You may need to edit all the run* files’ account information and other settings. However, if you belong to group s0492 then the scripts should work without any modifications.

Change account to appropriate SBU charge code:
#SBATCH --account s0942
Change if you want to change number of nodes, hasw - to run on haswell nodes:
#SBATCH --ntasks=16 --constraint=hasw
Uncomment and set according to your needs and privileges:
##SBATCH --qos=high
Uncomment (if desired) and substitute your e-mail here:
##SBATCH --mail-user=user@nasa.gov
A note about namelists settings

Things to keep in mind before we run NU-WRF components.

- The length of the simulations is specified in the namelist files:
  - In namelist.wps the length is determined by start_date and end_date
  - In namelist.input look for start_ and end_ fields.
  - The dates in both namelists must be consistent.
- The workflow is designed to work as-is. However, if you want to run for different dates:
  - You must get the corresponding atmospheric data for initial conditions.
  - You may need to modify the namelists. For example in namelist.input, make sure end_day - start_day = run_days.
- For any other changes please refer to the user’s guide.
GEOGRID

GEOGRID interpolates static and climatological terrestrial data (land use, albedo, vegetation greenness, etc) to each WRF grid.

- Input: namelist.wps
- Output: For $N$ domains (max_dom in namelist.wps), $N$ geo_em files will be created.

To run:

```
> cd $RUNDIR
> sbatch run_geogrid.bash
```

---

When done, check for "Successful completion" string in the file geogrid.slurm.out. geogrid.log.<node> will also be created for tracking run failures or debugging.
UNGRIB reads GRIB or GRIB2 files with dynamic meteorological and dynamic terrestrial data (soil moisture, soil temperature, sea surface temperature, sea ice, etc) and write specific fields in WPS intermediate format.

- **Input**: namelist.wps. Important fields are interval_seconds and start/end dates.
- **Output**: Several NAM* files corresponding to number of intervals (interval_seconds) in simulation length (start/end dates).

**NOTE**: The GRIB input is referenced in the run script, run_ungrib.bash:

```
./link_grib.csh data/ungrib/*
```

The UNGRIB output (NAM) is determined by the settings in the WPS namelist (namelist.wps).

To run:

```bash
> cd $RUNDIR
> ./run_ungrib.bash
``` 

---

When done, check for "Successful completion" string in the file ungrib.slurm.out. ungrib.log will also be created for tracking run failures or debugging.

To run:

> cd $RUNDIR
> mkdir RUN_SST2WRF
> cd RUN_SST2WRF
> cp $NUWRFDIR/utils/sst2wrf/scripts/Run_SST.csh .

For the specified (in namelist.wps) start date = 20090410 and end date of 20090411:

> ./Run_SST.csh 20090410 20090411 mw_ir . $NUWRFDIR

SSTRSS:* files will be created, and these files should be copied to the RUNDIR before running METGRID component.

> cp $RUNDIR/RUN_SST2WRF/sstdata/mw_ir/SSTRSS* $RUNDIR
METGRID horizontally interpolates UNGRIB and SSTRSS output to the WRF domains, and combine them with the output from GEOGRID.

- **Input:** namelist.wps, geo_em*, NAM*, and SSTRSS* files.
- **Output:** Several met_em* files corresponding to number of intervals (interval_seconds) in simulation length (start/end dates).

To run:

```bash
> cd $RUNDIR
> sbatch run_metgrid.bash
```

When done, check for "Successful completion" string in the file metgrid.slurm.out. metgrid.log.<node> will also be created for tracking run failures or debugging.
REAL vertically interpolates the METGRID output to the WRF grid, and creates initial and lateral boundary condition files.

- **Input:** namelist.input, met_em* files.
- **Output:** wrfinput* files (one for each domain), wrfbdy_d01.

To run:

```
> cd $RUNDIR
> sbatch run_real.bash
---
```

Check real.slurm.out for run completion.
Also check real.rsl.out.<node> and real.rsl.error.<node> for tracking run failures or debugging.
GOCART2WRF extracts aerosol data from GEOS-5 netCDF4 GOCART files (or MERRA reanalysis aerosol data files); horizontally and vertically interpolates the fields to the WRF grid; and appends them to the initial and lateral boundary condition files of WRF (wrfinput_d* and wrfbdy_d01). User can choose GOCART aerosol data or MERRA reanalysis Aerosol data.

- **Input:** namelist.gocart2wrf, grib_input/*, wrfinput* files (one for each domain), wrfbdy_d01.
- **Output:** wrfinput* files (one for each domain), wrfbdy_d01. Original wrfinput* and wrfbdy_d01 files will be backed up with .gocart2wrf extension.

To run:

```
> cd $RUNDIR
>./run_gocart2wrf.bash # runs in serial
```

Check this file for successful run completion: gc2wrf.log.
This community tool processes a number of biogenic, anthropogenic, volcanic, and wildfire emissions. The NU-WRF version has several modifications that are discussed in the user’s guide. Running prep Chem_sources requires a prep Chem_sources.inp file and upon completion it produces map projection data (that can be visualized by plot Chem) as well as other files used by convert emiss.

- **Input:** prep Chem_sources.inp
- **Output:** nuwrf-T* files

To run:

```bash
> cd $RUNDIR
> ./run_prep_chem_sources.bash
```

Check this file for successful run completion: pcs.log.
This is a community WRF-Chem preprocessor that takes the output from PREP CHEM SOURCES and rewrites the fields in new netCDF files for reading by WRF-Chem.

- Input: namelist.input.convert_emiss files (one for each domain).
- Output: wrfchemi_gocart_bg* and wrfchemi* files (one for each domain).

To run:

> cd $RUNDIR
> sbatch run_convert_emiss.bash

---

Check ce.slurm.out for run completion.
If we get to this point then we are ready to run WRF with chemistry.

- **Input:** wrfinput* files (one for each domain), wrfbdy_d01, wrfchemi_gocart_bg* and wrfchemi* files.
- **Output:** wrfout* and wrfrst* files (one for each domain).

---

To run:

```bash
> cd $RUNDIR
> sbatch run_wrf.bash
```

Check wrf.slurm.out for run completion. Also check wrf.rsl.out.<node> and wrf.rsl.error.<node> for tracking run failures/debugging.
Post-processing on Discover

Using NCVIEW:

WRF output files (NETCDF4) can be viewed using a special version of ncview installed on Discover:

/usr/local/other/SLES11/ncview/2.1.1/bin/ncview <filename>
Using RIP (NCAR graphics). Submit the `rip` job:

```
> cd $RUNDIR
> run_rip.bash # (or use sbatch)
> idt filename.cgm # Substitute actual filename
```

run_rip.bash will run ripdp_wrfarw and rip to generate NCAR Graphics cgm files.

`idt` is a NCAR Graphics executable in `$NCARG_ROOT/bin`

Sample RIP plot specification tables are in `$NUWRFDIR/scripts/rip`

and are looped through by run_rip.bash

See [http://www2.mmm.ucar.edu/wrf/users/docs/ripug.htm](http://www2.mmm.ucar.edu/wrf/users/docs/ripug.htm) for info on customizing plots with RIP.

Minor changes to run_rip.bash may be necessary.
The workflow just described can be modified to use MERRA/MERRA2 reanalysis atmospheric initial and lateral boundary conditions, RSS SST (Remote Sensing Systems SST product) along with LIS land surface model initial conditions (see diagram in the next page). Just replace the UNGRIB step by the MERRA2WRF step and add the LDT-LIS-LDT steps before running REAL.
Alternate NU-WRF chemistry workflow.

This NU-WRF-CHEM workflow uses MERRA/ MERRA2 reanalysis atmospheric initial and lateral boundary conditions, RSS SST (Remote Sensing Sytems SST product) along with LIS land surface model initial conditions, and other chemical tracers from Prep-chem sources, and runs WRF-CHEM.
SCM Workflow
The SCM workflow describes the steps necessary to setup and run WRF-LIS (like the default workflow) using a single point domain.

*No chemistry is used.*
NU-WRF Single Column Model (SCM) workflow.

New WRF-LIS Single Column Model workflow

Run WPS with 3x3 staggered grid centered on point of interest

WRF-LIS runs with horizontally homogeneous fields with unstaggered 2x2 grid

LIS and LDT run in 2x2 unstaggered grid. Point of interest should be southwest grid box
Copy the following files to **RUNDIR**:  

```bash  
> cd $RUNDIR  
> cp $PROJECTDIR/tutorial/scm/*.* .  
> cp -r $PROJECTDIR/tutorial/scm/data/ungrib ungrib_input  
```

Where:

- `common.cfg` : shared script with settings used by other scripts. 
- `run*.bash` : scripts to run pre-processors and model. 
- `namelist*` : namelist files required by executables.
Required script changes

> cd $RUNDIR

Use your favorite editor to edit `common.cfg` and change the values of NUWRFDIR and RUNDIR using the values set earlier.

# *** Please make sure these settings are correct ***
# NUWRFDIR specifies the location of the NU-WRF source code
NUWRFDIR=<CHANGE THIS>
# RUNDIR specifies the location of the temporary run directory
RUNDIR=<CHANGE THIS>

You may need to edit all the run* files’ account information and other settings. However, if you belong to group s0492 then the scripts should work without any modifications.

Change account to appropriate SBU charge code:
#SBATCH --account s0942
The SCM runs on 1 CPU only. Do not change --ntasks to any other value!
#SBATCH --ntasks=1
Most SCM runs are very short, so use debug queue if possible.
##SBATCH --qos=debug
Uncomment (if desired) and substitute your e-mail here:
##SBATCH --mail-user=user@nasa.gov
A note about namelists settings

Things to keep in mind before we run NU-WRF components.

- The length of the simulations is specified in the namelist files:
  - In namelist.wps the length is determined by start_date and end_date
  - In namelist.input look for start_ and end_ fields.
  - The dates in both namelists must be consistent.

- The workflow is designed to work as-is. However, if you want to run for different dates:
  - You must get the corresponding atmospheric data for initial conditions.
  - You may need to modify the namelists. For example in namelist.input, make sure end_day - start_day = run_days.

- For any other changes please refer to the user’s guide.
GEOGRID interpolates static and climatological terrestrial data (land use, albedo, vegetation greenness, etc) to each WRF grid.

- Input: namelist.wps
- Output: For \( N \) domains (max.dom in namelist.wps), \( N \) geo_em files will be created.

To run:

```bash
> cd $RUNDIR
> sbatch run_geogrid.bash
---
```

When done, check for "Successful completion" string in the file geogrid.slurm.out. geogrid.log.<node> will also be created for tracking run failures or debugging.
UNGRIB reads GRIB or GRIB2 files with dynamic meteorological and dynamic terrestrial data (soil moisture, soil temperature, sea surface temperature, sea ice, etc) and writes specific fields in a WPS intermediate format.

- **Input:** namelist.wps, ungrib_input/* files.
- **Output:** NARR* files.

**NOTE:** The GRIB input is referenced in the run script, run_ungrib.bash:

```bash
./link_grib.csh ungrib_input/*
```

The UNGRIB output (NARR) is determined by the settings in the WPS namelist (namelist.wps).

To run:

```bash
> cd $RUNDIR
> sbatch run_ungrib.bash
``` 

When done, check for "Successful completion" string in the file ungrib.slurm.out. ungrib.log will also be created for tracking run failures or debugging.
METGRID horizontally interpolates the output from UNGRIB to the WRF domains, and combines it with the output from GEOGRID.

- **Input:** namelist.wps, NARR* files, geo_em* files.
- **Output:** Several met_em* files corresponding to number of intervals (interval_seconds) in simulation length (start/end dates).

To run:

```bash
> cd $RUNDIR
> sbatch run_metgrid.bash
```  

When done, check for "Successful completion" string in the file metgrid.slurm.out. metgrid.log.<node> will also be created for tracking run failures or debugging.
BUILD_SCM_FORCING scripts use WPS output and generate column initial conditions profile_init.txt and surface_init.txt.
Note that this script depends on NCL and on Discover one needs to load other/ncl-6.3.0-static.

- **Input:** WPS output, forcing_file.cdl and several *.ncl files.
- **Output:** profile_init.txt, surface_init.txt, and a directory, 2006-07-14_00:00:00, containing forcing_file.nc and input_sounding data.

To run:

> cd $RUNDIR

Edit build_scm_forcing.bash: set metPath and forceArcRoot equal to RUNDIR value.

> build_scm_forcing.bash # runs in serial
LDT (pre-LIS)

LDT processes data inputs for different surface models. In this use-case we are using the Noah v3.6 land surface model.

- **Input**: ldt.config
- **Output**: lis_input* files for each NU-WRF domain.

To run:

```bash
> cd $RUNDIR
> sbatch run_ldt_prelis.bash
```

When done, check for "Finished LDT run" string in the file `ldt_log_prelis.0000`
LIS can be run from a cold start or from a restart. A cold start run is usually a multi-year simulation, not appropriate for a tutorial. In the restart case we have to provide restart files as input (LIS_RST* files) and LIS run is significantly shorter. Nevertheless, the restart files have already been generated for this tutorial and the user can **skip** this step.

- **Input:** lis.config, LIS_RST_NOAH36*nc, forcing_variables_wrfcplmode.txt, NOAH36_OUTPUT_LIST_SPINUP.TBL
- **Output:** LIS_RST_NOAH36*nc files, one for each NU-WRF domain.

**To run:**

```
> cd $RUNDIR
> sbatch run_lis.bash
---
```

Check this file for successful run completion: lis.slurm.out
lislog.<node> will also be created for tracking run failures or debugging.
After running LIS, it is necessary to rerun LDT in "NUWRF preprocessing for real" mode. This requires modifications to ldt.config to specify the static output file from LDT and the dynamic output file from LIS. Fields from both will be combined and written to a new netCDF output file for use by REAL

- **Input:** ldt.config
- **Output:** lis4real_input* files for each NU-WRF domain.

To run:

```
> cd $RUNDIR
> sbatch run_ldt_postlis.bash
---
When done, check for "Finished LDT run" string in the file
  ldt_log_postlis.0000
```
LIS4SCM copies data from southwest grid point to remainder of LIS/LDT domain (becomes horizontally homogenous): LIS4SCM imposes identical lat/lon at each LIS point in the restart file.

- Input: namelist.lis4scm, LDT-LIS-LDT output
- Output: Horizontally homogeneous lis_input file.

To run:

> cd $RUNDIR
> ./run_lis4scm.bash # Note this runs in serial
IDEAL creates wrfinput file from text column data and homogeneous LIS/LDT file.

- Input: LIS4SCM output
- Output: wrfinput file.

To run:

> cd $RUNDIR
> cp 2006-07-14_00:00:00/* .
> sbatch run_ideal.bash
WRF-LIS reads wrfinput and homogenous LIS restart file.

- **Input:** wrfinput file
- **Output:** wrfoutput file.

To run:

> cd $RUNDIR
> sbatch run_wrf.bash

Notes:

- Tested case uses periodic LBCs (thus, no wrfbdy file).
- Tested case also used no forcing (no external advection, etc).
- Tested case used dx = 1 km, dt = 6 sec (runs very fast!)
Using NCVIEW:

WRF output files (NETCDF4) can be viewed using a special version of ncview installed on Discover:

/usr/local/other/SLES11/ncview/2.1.1/bin/ncview <filename>
Using RIP (NCAR graphics). Submit the **rip** job:

```bash
> cd $RUNDIR
> run_rip.bash # (or use sbatch)
> idt filename.cgm # Substitute actual filename
```

run_rip.bash will run ripdp_wrfarw and rip to generate NCAR Graphics cgm files.

idt is a NCAR Graphics executable in `$NCARG_ROOT/bin`

Sample RIP plot specification tables are in `$NUWRFDIR/scripts/rip` and are looped through by run_rip.bash

See [http://www2.mmm.ucar.edu/wrf/users/docs/ripug.htm](http://www2.mmm.ucar.edu/wrf/users/docs/ripug.htm) for info on customizing plots with RIP.

Minor changes to run_rip.bash may be necessary.
Regression testing
Regression testing is a type of software testing that verifies that software previously developed and tested still performs correctly even after it was changed or interfaced with other software. Changes may include software enhancements, patches, configuration changes, etc. During regression testing, new software bugs or regressions may be uncovered.
Testing types

Software testing can be roughly divided into three categories:

- Automated regression tests. These tests compile and run a very large number of model configurations and may perform various checks.

- Manual/user testing. These tests are intended to be performed by users that wish to verify their changes prior to pushing to a central repository. In detail, these tests are similar to the automated tests, but are more easily executed from the command line.

- Unit tests. These tests execute extremely quickly and provide a finer-grained verification than the other regression tests, albeit with far less coverage of the source code. *Unit tests do not form part of the NU-WRF code base at this time.*

In this section we discuss the available NU-WRF regression testing infrastructure and how it can be used to perform automated and manual tests.
The NU-WRF regression scripts are a set of python scripts and configuration files designed to run tests on the NU-WRF code base. The scripts’ functionality is all driven by the settings specified in a single configuration file.

The configuration file is a text file with a particular structure readable by the python scripts and easily understood by humans too! The files are organized into sections, and each section contains name-value pairs for configuration data.
NU-WRF regression testing scripts

There is one configuration file required by the scripts - specified as a command line argument. The file can have any name but must have the '.cfg' extension. A sample configuration file (sample.cfg) is included $NUWRFDIR/scripts/regression that can be used for "routine" testing.

Note:

There are two additional files that are used specifically for NU-WRF testing: master.cfg and comp.cfg. These files should not be modified or used for anything other than as reference to build your own customized configuration.
In this document we have discussed 4 workflows: basic, default, chemistry and scm. For the purposes of regression testing workflows are categorized as follows:

- **wrf**: default WRF-ARW runs (without LIS) (basic)
- **wrfhis**: like wrf but with LIS-coupling (default and scm)
- **chem**: like wrf with chemistry (chemistry)
- **kpp**: like wrf with chemistry - uses KPP (chemistry)

This distinction is necessary to specify test cases in the configuration file. For a list of all tested NU-WRF configurations see:

```
$NUWRFDIR/scripts/regression/master.cfg
```
Python scripts

The python scripts are the following:

- `reg` : Main driver.
- `RegPool.py` : A class with functionality to parallelize tasks.
- `regression.py` : A driver script to execute the NU-WRK component tasks.
- `RegRuns.py` : A class that defines regression test run instances.
- `RegTest.py` : A class that defines regression test instances.
- `reg_tools.py` : Various tools used by the main drivers.
- `reg_utils.py` : Various utilities used by all the scripts.

The scripts can be used to perform both automated regression tests as well as ”manual testing”.
Manual testing

Before running the regression testing scripts make sure you duplicate the sample.cfg file and customize it to your needs. When ready, it is recommended that you obtain an interactive queue with as many cores as required by your NU-WRF test case.

Request an interactive queue on DISCOVER, for example:

```
salloc --ntasks=84 --time=1:00:00 --account=s0492 --constraint=hasw
```

Once you get the prompt

```
$ cd $NUWRFDIR/scripts/regression
```

To run the scripts, specify the configuration file name as an argument to reg:

```
$ reg <cfg_file> &
```

Note that the extension should not be specified.
Manual testing

Upon execution you will see some messages echoed to STDOUT but all the output will be logged to a file named <cfg_file>.LOG.

There will be other "log" files, one for each <workflow> and one for each <experiment>, that will be generated for each 'reg' run.

Workflow builds, a maximum of four, will be generated in <scratch_dir>/builds. Each build will have <workflow>.out and <workflow>.err files that will be useful to diagnose build errors.
Run logs will be generated in each run directory under <scratch_dir>/results. These will have the name <experiment>-regression.log and should also be very useful to diagnose test-case-specific run-time issues.

At the end of a "reg" an email test report will be emailed to the recipient specified in the mail_to field in USERCONFIG.

Note that all 'reg' runs are tagged with a unique time stamp corresponding to the date/time the run was executed.
As you may have figured out, all the workflows described earlier in this tutorial can be recreated with the appropriate configuration file and the testing scripts.

For more information about regression testing or any workflow described in this tutorial please feel free to contact any member of the NU-WRF software integration group.