Chapter 5: WRF Model

Table of Contents

Introduction

The WRF model is a fully compressible and nonhydrostatic model (with a run-time hydrostatic option). Its vertical coordinate is a terrain-following hydrostatic pressure coordinate. The grid staggering is the Arakawa C-grid. The model uses the Runge-Kutta 2nd and 3rd order time integration schemes, and 2nd to 6th order advection schemes in both the horizontal and vertical. It uses a time-split small step for acoustic and gravity-wave modes. The dynamics conserves scalar variables.

The WRF model code contains an initialization program (either for real-data, *real.exe*, or idealized data, *ideal.exe*; see Chapter 4), a numerical integration program (*wrf.exe*), a program to do one-way nesting (*ndown.exe*), and a program to do tropical storm bogussing (*tc.exe*). The WRF model, Version 3, supports a variety of capabilities. These include

- Real-data and idealized simulations
- Various lateral boundary condition options for real-data and idealized simulations
- Full physics options, and various filter options
- Positive-definite advection scheme
- Non-hydrostatic and hydrostatic (runtime option)
- One-way and two-way nesting, and a moving nest
- Three-dimensional analysis nudging
- Observation nudging
- Regional and global applications
- Digital filter initialization

Other References

- WRF tutorial presentation: http://www.mmm.ucar.edu/wrf/users/supports/tutorial.html
- WRF-ARW Tech Note: http://www.mmm.ucar.edu/wrf/users/pub-doc.html
- See chapter 2 of this document for software requirement.

Installing WRF

Before compiling the WRF code on a computer, check to see if the netCDF library is installed. This is because one of the supported WRF I/O options is netCDF, and it is the one commonly used and supported by the post-processing programs. If the netCDF is installed in a directory other than `/usr/local/`, then find the path, and use the environment variable NETCDF to define where the path is. To do so, type

```
setenv NETCDF path-to-netcdf-library
```

Often the netCDF library and its include/ directory are collocated. If this is not the case, create a directory, link both netCDF lib and include directories in this directory, and use the environment variable to set the path to this directory. For example,

```
netcdf_links/lib -> /netcdf-lib-dir/lib
netcdf_links/include -> /where-include-dir-is/include

setenv NETCDF /directory-where-netcdf_links-is/netcdf_links
```

If the netCDF library is not available on the computer, it needs to be installed first. NetCDF source code or pre-built binary may be downloaded from, and installation instruction can be found on, the Unidata Web page at http://www.unidata.ucar.edu/.

***Hint*:** for Linux users:

If PGI, Intel, gfortran or g95 compilers are are used on a Linux computer, make sure netCDF is installed using the same compiler. Use the NETCDF environment variable to point to the PGI/Intel/g95 compiled netCDF library.

Hint: If using netCDF-4, make sure that the new capabilities (such as parallel I/O based on HDF5) are not activated at the install time.

The WRF source code tar file can be downloaded from http://www.mmm.ucar.edu/wrf/users/download/get_source.html. Once the tar file is unzipped (`gunzip WRFV3.TAR.gz`), and untared (`tar -xf WRFV3.TAR`), it will create a WRFV3/ directory. This contains:

`Makefile`	Top-level makefile

`README`	General information about the WRF/ARW core
`README_test_cases`	Explanation of the test cases
`README.NMM`	General information for the WRF/NMM core
`README.rsl_output`	For NMM
`Registry/`	Directory for WRF Registry files
`arch/`	Directory where compile options are gathered
`clean`	script to clean created files and executables
`compile`	script for compiling the WRF code
`configure`	script to create the *configure.wrf* file for compiling
`chem/`	WRF chemistry, supported by NOAA/GSD
`dyn_em/`	Directory for ARW dynamics and numerics
`dyn_exp/`	Directory for a 'toy' dynamic core
`dyn_nmm/`	Directory for NMM dynamics and numerics, supported by DTC
`external/`	Directory that contains external packages, such as those for IO, time keeping and MPI
`frame/`	Directory that contains modules for the WRF framework
`inc/`	Directory that contains 'include' files
`main/`	Directory for main routines, such as wrf.F, and all executables after compilation
`phys/`	Directory for all physics modules
`run/`	Directory where one may run WRF
`share/`	Directory that contains mostly modules for the WRF mediation layer and WRF I/O
`test/`	Directory that contains test case directories, may be used to run WRF
`tools/`	Directory that contains tools for developers

The steps to compile and run the model are:

1. configure: generate a configuration file for compilation
2. compile: compile the code
3. run the model

Go to the WRFV3 (top) directory and type

```
./configure
```

and a list of choices for your computer should appear. These choices range from compiling for a single processor job (serial), to using OpenMP shared-memory (smpar) or distributed-memory parallelization (dmpar) options for multiple processors, or a combination of shared-memory and distributed-memory options (dm+sm). When a selection is made, a second choice for compiling nesting will appear. For example, on a Linux computer, the above steps may look like:

```
> setenv NETCDF /usr/local/netcdf-pgi
> ./configure

checking for perl5... no
checking for perl... found /usr/bin/perl (perl)
```

```
Will use NETCDF in dir: /usr/local/netcdf-pgi
PHDF5 not set in environment. Will configure WRF for use without.
$JASPERLIB or $JASPERINC not found in environment, configuring to build without grib2
I/O...
------------------------------------------------------------------------
Please select from among the following supported platforms.

1.  Linux i486 i586 i686, gfortran compiler with gcc  (serial)
2.  Linux i486 i586 i686, gfortran compiler with gcc  (smpar)
3.  Linux i486 i586 i686, gfortran compiler with gcc  (dmpar)
4.  Linux i486 i586 i686, gfortran compiler with gcc  (dm+sm)
5.  Linux i486 i586 i686, g95 compiler with gcc  (serial)
6.  Linux i486 i586 i686, g95 compiler with gcc  (dmpar)
7.  Linux i486 i586 i686, PGI compiler with gcc  (serial)
8.  Linux i486 i586 i686, PGI compiler with gcc  (smpar)
9.  Linux i486 i586 i686, PGI compiler with gcc  (dmpar)
10.  Linux i486 i586 i686, PGI compiler with gcc  (dm+sm)
11.  Linux x86_64 i486 i586 i686, ifort compiler with icc (non-SGI installations)
(serial)
12.  Linux x86_64 i486 i586 i686, ifort compiler with icc (non-SGI installations)
(smpar)
13.  Linux x86_64 i486 i586 i686, ifort compiler with icc (non-SGI installations)
(dmpar)
14.  Linux x86_64 i486 i586 i686, ifort compiler with icc (non-SGI installations)
(dm+sm)
15.  Linux i486 i586 i686 x86_64, PathScale compiler with pathcc (serial)
16.  Linux i486 i586 i686 x86_64, PathScale compiler with pathcc  (dmpar)

Enter selection [1-16] : 9

Compile for nesting? (0=no nesting, 1=basic, 2=preset moves, 3=vortex following) [default
0]: 1
```

Enter the appropriate options that are best for your computer and application.

When the return key is hit, a `configure.wrf` file will be created. Edit compile options/paths, if necessary.

Hint: It is helpful to start with something simple, such as the serial build. If it is successful, move on to build smpar or dmpar code. Remember to type 'clean –a' between each build.

Hint: If you anticipate generating a netCDF file that is larger than 2Gb (whether it is a single- or multi-time period data [e.g. model history]) file), you may set the following environment variable to activate the large-file support option from netCDF (in c-shell):

```
setenv WRFIO_NCD_LARGE_FILE_SUPPORT 1
```

Hint: If you would like to use parallel netCDF (p-netCDF) developed by Argonne National Lab (http://trac.mcs.anl.gov/projects/parallel-netcdf), you will need to install p-netCDF separately, and use the environment variable PNETCDF to set the path:

```
setenv PNETCDF path-to-pnetcdf-library
```

To compile the code, type

```
./compile
```

and the following choices will appear:

Usage:

```
compile wrf         compile wrf in run dir (Note, no real.exe, ndown.exe or ideal.exe generated)

or choose a test case (see README_test_cases for details):

        compile em_b_wave

        compile em_esmf_exp (example only)
        compile em_grav2d_x

        compile em_heldsuarez
        compile em_hill2d_x

        compile em_les
        compile em_quarter_ss
        compile em_real

        compile em_seabreeze2d_x
        compile em_squall2d_x
        compile em_squall2d_y
        compile em_tropical_cyclone

        compile exp_real (example of a toy solver)

        compile nmm_real (NMM solver)

compile –h          help message
```

where **em** stands for the Advanced Research WRF dynamic solver (which currently is the '**E**ulerian **m**ass-coordinate' solver). Type one of the above to compile. When you switch from one test case to another, you must type one of the above to recompile. The recompile is necessary to create a new initialization executable (i.e. `real.exe`, and `ideal.exe` - there is a different `ideal.exe` for each of the idealized test cases), while `wrf.exe` is the same for all test cases.

If you want to remove all object files (except those in the `external/` directory) and executables, type '*`clean`*'.

Type '*`clean -a`*' to remove built files in ALL directories, including `configure.wrf` (the original `configure.wrf` will be saved to `configure.wrf.backup`). This is recommended if you make any mistake during the process, or if you have edited the `configure.wrf` or `Registry.EM` file.

Hint: If you have trouble compiling routines, like solve_em.F, you can try to run the *`configure`* script with the optional argument '`-s`', i.e.

```
./configure -s
```

This will configure to compile solve_em.F and a few other routines with reduced optimization.

If you would like to turn off optimization for all the code, say during code development and debugging, you can run the *`configure`* script with option '*`-d`*':

```
./configure -d
```

a. Idealized case

For any 2D test case (labeled in the case names), serial or OpenMP (smpar) compile options must be used. Additionally, you must only choose the '0=no nesting' option when you configure. For all other cases, you may use serial or parallel (dmpar) and nesting. Suppose you would like to compile and run the 2-dimensional squall case, type

```
./compile em_squall2d_x >& compile.log
```

After a successful compilation, you should have two executables created in the **main/** directory: **ideal.exe** and **wrf.exe**. These two executables will be linked to the corresponding test/case_name and run/ directories. cd to either directory to run the model.

It is a good practice to save the entire compile output to a file. When the executables are not present, this output is useful to help diagnose the compile errors.

b. Real-data case

For a real-data case, type

```
./compile em_real >& compile.log &
```

When the compile is successful, it will create three executables in the **main**/directory: ndown.exe**,** real.exe and wrf.exe.

real.exe: for WRF initialization of real data cases
ndown.exe : for one-way nesting
wrf.exe : WRF model integration

Like in the idealized cases, these executables will be linked to the test/em_real and run/ directories. cd to one of these two directories to run the model.

Running WRF

One may run the model executables in either the run/ directory, or the test/case_name directory. In either case, one should see executables ideal.exe or real.exe (and ndown.exe), and wrf.exe, linked files (mostly for real-data cases), and one or more namelist.input files in the directory.

Hint: If you would like to run the model executables in a different directory, copy or link the files in the test/em_* directory to that directory, and run from there.

a. Idealized case

Suppose the test case em_squall2d_x is compiled. To run, type

```
cd test/em_squall2d_x
```

Edit the `namelist.input` file (see `README.namelist` in the `WRFV3/run/` directory or its Web version) to change length of integration, frequency of output, size of domain, timestep, physics options, and other parameters.

If you see a script in the test case directory, called `run_me_first.csh`, run this one first by typing:

```
./run_me_first.csh
```

This links some physics data files that might be needed to run the case.

*Note: when running *em_fire*, you must copy everything from the 'hill_simple' directory into your current working directory in order for it to run correctly.

```
cp hill_simple/* .
```

To run the initialization program, type

```
./ideal.exe
```

This program will typically read an input sounding file located in that directory, and generate an initial condition file `wrfinput_d01`. All idealized cases do not require a lateral boundary file because of the boundary condition choices they use, such as the periodic option. If the job is run successfully, the last thing it prints should be: '`wrf: SUCCESS COMPLETE IDEAL INIT`'.

To run the model and save the standard output to a file, type

```
./wrf.exe >& wrf.out &
```

or for a 3D test case compiled with MPI (dmpar) option,

```
mpirun -np 4 ./wrf.exe
```

If successful, the wrf output file will be written to a file named `wrfout_d01_0001-01-01_00:00:00`.

Pairs of `rsl.out.*` and `rsl.error.*` files will appear with any MPI runs. These are standard out and error files. Note that the execution command for MPI runs may be different on different machines and for different MPI installation. Check the user manual.

If the model run is successful, the last thing printed in the '`wrf.out`' or `rsl.*.0000` files should be: '`wrf: SUCCESS COMPLETE WRF`'. Output files `wrfout_d01_0001-01-01*` and `wrfrst*` should be present in the run directory, depending on how namelist variables are specified for output. The time stamp on these files originates from the start times in the namelist file.

b. Real-data case

To make a real-data case run, `cd` to the working directory by typing

```
cd test/em_real (or cd run)
```

Start with the `namelist.input` template file in the directory and edit it to match your case.

Running a real-data case requires successfully running the **WRF Preprocessing System** programs (or WPS).

Make sure `met_em.*` files from WPS are seen in the run directory (either link or copy the files):

```
cd test/em_real
ls -l ../../../WPS/met_em*
ln -s ../../..WPS/met_em* .
```

Make sure you edit the following variables in the `namelist.input` file:

`num_metgrid_levels`: number of incoming data levels (can be found by using the `ncdump` command on the `met_em.*` file)
`num_metgrid_soil_levels`: number of incoming soil data levels
`eta_levels`: model *eta* levels from 1 to 0, if you choose to do so. If not, `real` will compute a nice set of *eta* levels. The computed eta levels have 7 half levels in the lowest 1 km or so, and stretches to constant δz.

Other options for use to assist vertical interpolation are:

`use_surface`: whether to use surface input data
`extrap_type`: vertical extrapolation of non-temperature fields
`t_extrap_type`: vertical extrapolation for potential temperature
`use_levels_below_ground`: use levels below the input surface level
`force_sfc_in_vinterp`: force vertical interpolation to use surface data
`lowest_lev_from_sfc`: place surface data in the lowest model level
`p_top_requested`: pressure top used in the model, default is 5000 Pa
`interp_type`: vertical interpolation method: linear in p(default) or log(p)
`lagrange_order`: vertical interpolation order, linear (default) or quadratic
`zap_close_levels`: allow surface data to be used if it is close to a constant pressure level.
`smooth_cg_topo`: smooth topography on the outer rows and columns in domain 1.
`use_tavg_for_tsk`: whether to use diurnally-averaged surface temp as skin temp. The diurnally- averaged surface temp can be computed using the WPS utility `avg_tsfc.exe`. This option can be used when SKINTEMP is not present.

Other minimum set of namelist variables to edit are:

`start_*, end_*`: start and end times for data processing and model integration
`interval_seconds`: input data interval for boundary conditions
`time_step`: model time step, and can be set as large as 6*DX (in km)
`e_ws, e_sn, e_vert`: domain dimensions in west-east, south-north and vertical
`dx, dy`: model grid distance in meters

To run the real-data initialization program, compiled using serial or OpenMP (smpar) options, type

```
./real.exe >& real.out
```

Successful completion of the job should have 'real_em: SUCCESS EM_REAL INIT' printed at the end of the real.out file. It should also produce wrfinput_d01 and wrfbdy_d01 files. In the real data case, both files are required.

Run the WRF model by typing

```
./wrf.exe
```

A successful run should produce one or several output files with names like `wrfout_d<domain>_<date>` (where `<domain>` represents domain ID, and `<date>` represents a date string with the format

`yyyy-mm-dd_hh:mm:ss`. For example, if you start the model at 1200 UTC, January 24 2000, then your first output file should have the name:

```
wrfout_d01_2000-01-24_12:00:00
```

The time stamp on the file name is always the first time the output file is written. It is always good to check the times written to the output file by typing:

```
ncdump -v Times wrfout_d01_2000-01-24_12:00:00
```

You may have other wrfout files, depending on the namelist options (how often you split the output files by using the namelist option `frames_per_outfile`). You may also create restart files if you have a restart frequency (`restart_interval` in the namelist.input file) set within your total integration time. The restart file should have names like

```
wrfrst_d<domain>_<date>
```

The time stamp on a restart file is the time at which that restart file is valid.

For DM (distributed memory) parallel systems, some form of the **mpirun** command will be needed to run the executables. For example, on a Linux cluster, the command to run MPI code, using 4 processors, may look like:

```
mpirun -np 4 ./real.exe
mpirun -np 4 ./wrf.exe
```

On some IBMs, the command for a batch job may be:

```
poe ./real.exe
poe ./wrf.exe
```

or

`mpirun.lsf ./wrf.exe` (on NCAR IBM *bluefire*)

c. Restart Run

A restart run allows a user to extend a run to a longer simulation period. It is effectively a continuous run made of several shorter runs. Hence the results at the end of one or more restart runs should be identical to a single run without any restart.

In order to do a restart run, one must first create a restart file. This is done by setting the namelist variable `restart_interval` (unit is in minutes) to be equal to or less than the simulation length in the first model run, as specified by `run_*` variables or `start_*` and `end_*` times. When the model reaches the time to write a restart file, a restart file named `wrfrst_d<domain>_<date>` will be written. The date string represents the time when the restart file is valid.

When one starts the restart run, edit the `namelist.input` file, so that your `start_*` time will be set to the restart time (which is the time the restart file is written). The other namelist variable one must set is `restart`, this variable should be set to .true. for a restart run.

In summary, these namelists should be modified:

`start_*, end_*:`	start and end times for restart model integration
`restart:`	logical to indicate whether the run is a restart or not

If the history and restart intervals are changed in a restart run, and the outcome isn't what is expected to be, use namelist '`override_restart_timers = .true.`'.

If history output is desired at the time of restart, use namelist '`write_hist_at_0h_rst = .true.`'

***Hint*:** Typically the restart file is a lot bigger in size than the history file, hence one may find that it is even ok to write a single model history output time to a file in netCDF format (`frame_per_outfile=1`), but it may fail to write a restart file. This is because the basic netCDF file support is only 2Gb. There are two solutions to the problem. The first is to simply set the namelist option `io_form_restart = 102` (instead of 2), and this will force the restart file to be written into multiple pieces, one per processor. As long as one restarts the model using the same number of processors, this option works well (and one should restart the model with the same number of processors in any case). The second solution is to recompile the code using the netCDF large file support option (see the section on "Installing WRF" in this chapter).

d. Two-way Nested Runs

A two-way nested run is a run in which multiple domains at different grid resolutions are run simultaneously and communicate with each other: The coarser domain provides boundary values for the nest, and the nest feeds its calculation back to the coarser domain. The model can handle multiple domains at the same nest level (no overlapping nest), and multiple nest levels (telescoping).

When preparing for a nested run, make sure that the code is compiled with basic nest options (option 1).

Most of options to start a nest run are handled through the namelist. ***All variables in the*** `namelist.input` ***file that have multiple columns of entries need to be edited with caution***. Start with a namelist template. The following are the key namelist variables to modify:

`start_*, end_*`: start and end simulation times for the nest

`input_from_file`: whether a nest requires an input file (e.g. `wrfinput_d02`). This is typically used for a real data case, since the nest input file contains nest topography and land information.

`fine_input_stream`: which fields from the nest input file are used in nest initialization. The fields to be used are defined in the Registry.EM. Typically they include static fields (such as terrain and landuse), and masked surface fields (such as skin temperature, soil moisture and temperature). Useful for a nest starting at a later time than the coarse domain.

`max_dom`: the total number of domains to run. For example, if you want to have one coarse domain and one nest, set this variable to 2.

`grid_id`: domain identifier that is used in the wrfout naming convention. The most coarse grid must have grid_id of 1.

`parent_id`: used to indicate the parent domain of a nest. `grid_id` value is used.

`i_parent_start/j_parent_start`: lower-left corner starting indices of the nest domain in its parent domain. These parameters should be the same as in `namelist.wps`.

`parent_grid_ratio`: integer parent-to-nest domain grid size ratio. Typically an odd number ratio is used in real-data applications.

`parent_time_step_ratio`: integer time-step ratio for the nest domain. It may be different from the `parent_grid_ratio`, though they are typically set the same.

`feedback`: this is the key setup to define a two-way nested (or one-way nested) run. When feedback is on, the

values of the coarse domain are overwritten by the values of the variables (average of cell values for mass points, and average of the cell-face values for horizontal momentum points) in the nest at the coincident points. For masked fields, only the single point value at the collocating points is fed back. If the `parent_grid_ratio` is even, an arbitrary choice of the southwest corner point value is used for feedback. This is the reason it is better to use an odd `parent_grid_ratio` with this option. When feedback is off , it is equivalent to a one-way nested run, since nest results are not reflected in the parent domain.

`smooth_option`: this a smoothing option for the parent domain in the area of the nest if *feedback* is on. Three options are available: 0 = no smoothing; 1 = 1-2-1 smoothing; 2 = smoothing-desmoothing.

3-D Idealized Cases

For 3-D idealized cases, no nest input files are required. The key here is the specification of the `namelist.input` file. What the model does is to interpolate all variables required in the nest from the coarse domain fields. Set

input_from_file = T, F,

Real Data Cases

For real-data cases, three input options are supported. The first one is similar to running the idealized cases. That is to have all fields for the nest interpolated from the coarse domain (`input_from_file = T, F`). The disadvantage of this option is obvious: one will not benefit from the higher resolution static fields (such as terrain, landuse, and so on).

The second option is to set `input_from_file = T` for each domain, which means that the nest will have a nest wrfinput file to read in. The limitation of this option is that this only allows the nest to start at the same time as the coarse domain.

The third option is, in addition to setting `input_from_file = T` for each domain, also set `fine_input_stream = 2` for each domain. Why a value of 2? This is based on the Registry setting, which designates certain fields to be read in from the auxiliary input stream number 2. This option allows the nest initialization to use 3-D meteorological fields interpolated from the coarse domain, static fields and masked, and time-varying surface fields from the nest wrfinput; hence it allows a nest to start at a later time than hour 0. Setting `fine_input_stream = 0` is equivalent to the second option.

To run `real.exe` for a nested run, one must first run WPS and create data for all the nests. Suppose WPS is run for a 24 hour period, two-domain nested case starting at 1200 UTC Jan 24 2000. Then the following files should be generated in a WPS directory:

```
met_em.d01.2000-01-24_12:00:00
met_em.d01.2000-01-24_18:00:00
met_em.d01.2000-01-25_00:00:00
met_em.d01.2000-01-25_06:00:00
met_em.d01.2000-01-25_12:00:00
met_em.d02.2000-01-24_12:00:00
```

Typically only the first time period of the nest input file is needed to create a nest wrfinput file. Link or move all these files to the run directory.

Edit the `namelist.input` file and set the correct values for all relevant variables, described on the previous pages (in particular, set `max_dom = 2`, for the total number of domains to run), as well as physics options. Type the following to run:

```
./real.exe >& real.out
```
or
```
mpirun -np 4 ./real.exe
```

If successful, this will create all input files for coarse, as well as nested domains. For a two-domain example, these are created:

```
wrfinput_d01
wrfinput_d02
wrfbdy_d01
```

To run WRF, type

```
./wrf.exe
```
or
```
mpirun -np 4 ./wrf.exe
```

If successful, the model should create wrfout files for both domain 1 and 2:

```
wrfout_d01_2000-01-24_12:00:00
wrfout_d02_2000-01-24_12:00:00
```

e. One-way Nested Run Using ndown

WRF supports two separate one-way nested options. In this section, one-way nesting is defined as a finer-grid-resolution run, made as a subsequent run after the coarser-grid-resolution run, where the `ndown` program is run in-between the two simulations. The initial and lateral boundary conditions for this finer-grid run are obtained from the coarse grid run, together with input from higher resolution terrestrial fields (e.g. terrain, landuse, etc.), and masked surface fields (such as soil temperature and moisture). The program that performs this task is `ndown.exe`. Note that the use of this program requires the code to be compiled for nesting.

When one-way nesting is used, the coarse-to-fine grid ratio is only restricted to be an integer. An integer less than or equal to 5 is recommended. Frequent output (e.g. hourly) from the coarse grid run is also recommended to provide better boundary specifications.

A caveat with using `ndown` for one-way nesting is that the microphysics variables are not used for boundary conditions; they are only in the initial conditions. If that is important to you, use the two-way nesting option instead.

Making a one-way nested run involves these steps:

1) Generate a coarse-grid model output
2) Make a temporary fine-grid initial condition `wrfinput_d01` file (note that only a single time period is required, valid at the desired start time of the fine-grid domain).
3) Run the program `ndown`**,** with coarse-grid model output and a fine-grid initial condition to generate fine grid initial and boundary conditions (similar to the output from the `real.exe` program).
4) Run the fine-grid simulation.

To compile, choose an option that supports nesting.

Before running, WPS should be run for one coarse and one nested domain (this helps to line-up the nest with the coarse domain), and for the one time period in which the one-way nested run is to start. This generates a WPS output file for the nested domain (domain 2): `met_em.d02.<date>`.

Step 1: Make a coarse grid run.

This is no different than any of the single-domain WRF runs, as described above.

Step 2: Make a fine-grid initial condition file.

The purpose of this step is to ingest higher resolution terrestrial fields and corresponding land-water masked soil fields.

- Rename `met_em.d02.*` to `met.d01.*` for the single requested fine-grid start time. Move the original domain 1 WPS output files before you do this.
- Edit the `namelist.input` file for the fine-grid domain (pay attention to column 1 only) and edit in the correct start time and grid dimensions.
- Run `real.exe` for this domain. This will produce a `wrfinput_d01` file.
- Rename this `wrfinput_d01` file to `wrfndi_d02`.

Step 3: Make the final fine-grid initial and boundary condition files

- Edit `namelist.input` again, and this time one needs to edit two columns: one for dimensions of the coarse grid, and one for the fine grid. Note that the boundary condition frequency (the namelist variable `interval_seconds`) is the time in seconds between the coarse-grid model output times. Since V3.2, one must also specify `io_form_auxinput2` = 2 to run `ndown` successfully.
- Run `ndown.exe`, with inputs from the coarse grid `wrfout` file(s), and the `wrfndi_d02` file generated from Step 2 above. This will produce the `wrfinput_d02` and `wrfbdy_d02` files.
- If one desires to refine the vertical resolution when running `ndown`, set `vert_refine_fact =` *`integer`* (new in V3.2). There are no other changes required in the namelist or in the procedure.
- Another way to refine vertical resolution is to use the utility program `v_interp` (see the chapter for 'Utilities and Tools' for details).

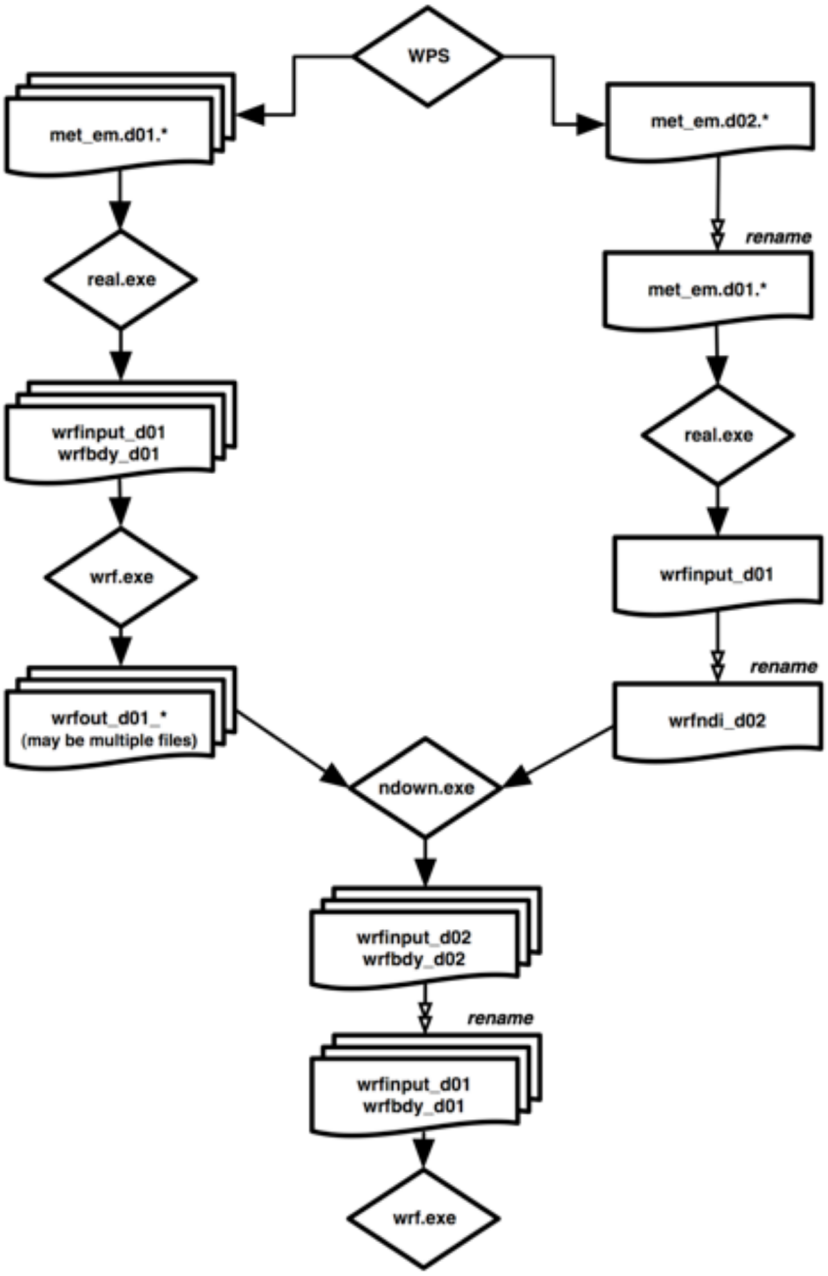
Note that the program `ndown` may be run serially or in MPI, depending on the selected compile option. The `ndown` program must be built to support nesting, however. To run the program, type

```
./ndown.exe
```
or
```
mpirun -np 4 ./ndown.exe
```

Step 4: Make the fine-grid WRF run

- Rename `wrfinput_d02` and `wrfbdy_d02` to `wrfinput_d01` and `wrfbdy_d01`, respectively.
- Edit `namelist.input` one more time, and it is now for the fine-grid domain only.
- Run WRF for this grid.

The figure on the next page summarizes the data flow for a one-way nested run using the program `ndown`.


WPS
met_em.d01.*
met_em.d02.*
rename
real.exe
met_em.d01.*
wrfinput_d01
wrfbdy_d01
real.exe
wrf.exe
wrfinput_d01
rename
wrfout_d01_*
(may be multiple files)
wrfndi_d02
ndown.exe
wrfinput_d02
wrfbdy_d02
rename
wrfinput_d01
wrfbdy_d01
wrf.exe

f. Moving-Nested Run

Two types of moving tests are allowed in WRF. In the first option, a user specifies the nest movement in the namelist. The second option is to move the nest automatically, based on an automatic vortex-following algorithm. This option is designed to follow the movement of a well-defined tropical cyclone.

To make the specified moving nested run, select the right nesting compile option (option 'preset moves'). Note that code compiled with this option will not support static nested runs. To run the model, only the coarse grid input files are required. In this option, the nest initialization is defined from the coarse grid data - no nest input is used. In addition to the namelist options applied to a nested run, the following needs to be added to the namelist section `&domains:`

`num_moves`: the total number of moves one can make in a model run. A move of any domain counts against this total. The maximum is currently set to 50, but it can be changed by changing MAX_MOVES in `frame/module_driver_constants.F`.

`move_id`: a list of nest IDs, one per move, indicating which domain is to move for a given move.

`move_interval`: the number of minutes from the beginning of the run until a move is supposed to occur. The nest will move on the next time step after the specified instant of model time has passed.

`move_cd_x,move_cd_y`: distance in the number of grid points and direction of the nest move (positive numbers indicate moving toward east and north, while negative numbers indicate moving toward west and south).

Parameter `max_moves` is set to be 50, but can be modified in the source code file *frame/module_driver_constants.F,* if needed.

To make the automatic moving nested runs, select the 'vortex-following' option when configuring. Again note that this compile would only support the auto-moving nest, and will not support the specified moving nested run or static nested run at the same time. Again, no nest input is needed. If one wants to use values other than the default ones, add and edit the following namelist variables in the `&domains` section:

`vortex_interval`: how often the vortex position is calculated in minutes (default is 15 minutes).

`max_vortex_speed`: used with *vortex_interval* to compute the search radius for the new vortex center position (default is 40 m/sec).

`corral_dist`: the distance in the number of coarse grid cells that the moving nest is allowed to get near the mother domain boundary (default is 8). This parameter can be used to center the telescoped nests so that all nests are moved together with the storm.

`track_level`: the pressure level (in Pa) where the vortex is tracked.

`time_to_move`: the time (in minutes) to move a nest. This option may help with the case when the storm is still too weak to be tracked by the algorithm.

When the automatic moving nest is employed, the model dumps the vortex center location, with minimum mean sea-level pressure and maximum 10-m winds in a standard-out file (e.g. `rsl.out.0000`). Typing '`grep ATCF rsl.out.0000`' will produce a list of storm information at a 15-minute interval:

```
ATCF    2007-08-20_12:00:00           20.37   -81.80     929.7 133.9
ATCF    2007-08-20_12:15:00           20.29   -81.76     929.3      133.2
```

In both types of moving-nest runs, the initial location of the nest is specified through `i_parent_start` and

j_parent_start in the namelist.input file.

The automatic moving nest works best for a well-developed vortex.

g. Analysis Nudging Runs (Upper-Air and/or Surface)

Prepare input data to WRF as usual using WPS. If nudging is desired in the nest domains, make sure all time periods for all domains are processed in WPS. For surface-analysis nudging (new in Version 3.1), OBSGRID needs to be run after METGRID, and it will output a wrfsfdda_d01 file that the WRF model reads for this option.

Set the following options before running real.exe, in addition to others described earlier (see the namelists in examples.namelist in the test/em_real/ directory, for guidance):

```
grid_fdda = 1
grid_sfdda = 1
```

Run real.exe as before, and this will create, in addition to wrfinput_d0* and wrfbdy_d01 files, a file named 'wrffdda_d0*'. Other grid-nudging namelists are ignored at this stage, but it is good practice to fill them all in before one runs real. In particular, set

```
gfdda_inname    =  "wrffdda_d<domain>"
gfdda_interval  = time interval of input data in minutes
gfdda_end_h     = end time of grid-nudging in hours

sgfdda_inname    =  "wrfsfdda_d<domain>"
sgfdda_interval  = time interval of input data in minutes
sgfdda_end_h     = end time of surface grid-nudging in hours
```

See http://www.mmm.ucar.edu/wrf/users/wrfv3.1/How_to_run_grid_fdda.html and README.grid_fdda in WRFV3/test/em_real/ for more information.

Spectral Nudging is a new upper-air nudging option since Version 3.1. This selectively nudges the coarser scales only, but is otherwise set up the same way as grid-nudging. This option also nudges geopotential height. The wave numbers defined here are the number of waves contained in the domain, and the number is the maximum one that is nudged.

```
grid_fdda = 2
xwavenum = 3
ywavenum = 3
```

h. Observation Nudging Run

In addition to the usual input data preparation using WPS, station observation files are required. See http://www.mmm.ucar.edu/wrf/users/wrfv3.1/How_to_run_obs_fdda.html for instructions. The observation file names expected by WRF are OBS_DOMAIN101 for domain 1, and OBS_DOMAIN201 for domain 2, etc.

Observation nudging is activated in the model by the following namelists in &fdda:

`obs_nudge_opt` = 1
`fdda_start` = 0 (obs nudging start time in minutes)
`fdda_end` = 360 (obs nudging end time in minutes)

and in `&time_control`

`auxinput11_interval_s` = 180, 180, 180, (set the interval to be small enough so that all observations will be checked)

Look for an example to set other obs nudging namelist variables in the file `examples.namelists` in `test/em_real/` directory. See http://www.mmm.ucar.edu/wrf/users/wrfv3.1/How_to_run_obs_fdda.html and `README.obs_fdda` in `WRFV3/test/em_real/` for more information.

i. Global Run

WRFV3 supports global capability. To make a global run, run WPS, starting with the namelist template `namelist.wps.gloabl`. Set `map_proj = 'lat-lon',` and grid dimensions `e_we` and `e_sn` without setting `dx` and `dy` in `namelist.wps`. The `geogrid` program will calculate grid distances, and their values can be found in the global attribute section of `geo_em.d01.nc` file. Type `ncdump -h geo_em.d01.nc` to find out the grid distances, which will be needed in filling out WRF's `namelist.input` file. Grid distances in x and y directions may be different, but it is best that they are set similarly or the same. WRF and WPS assume the earth is a sphere, and its radius is 6370 km. There are no restrictions on what to use for grid dimensions, but for effective use of the polar filter in WRF, the east-west dimension should be set to $2^P*3^Q*5^R+1$ (where P, Q, and R are any integers, including 0).

Run the rest of the WPS programs as usual but only for one time period. This is because the domain covers the entire globe, and lateral boundary conditions are no longer needed.

Run the program `real.exe` as usual and for one time period only. The lateral boundary file `wrfbdy_d01` is not needed.

Copy `namelist.input.global` to `namelist.input`, and edit it. Run the model as usual.

Note: since this is a uncommon option in the model, use it with caution. Not all options have been tested. For example, all filter options have not been tested, and positive-definite options are not working for a lat-lon grid.

As an extension to the global lat-lon grid, the regional domain can also be set using a lat-lon grid. To do so, one needs to set both grid dimensions, and grid distances in degrees. Again `geogrid` will calculate the grid distance, assuming the earth is a sphere and its radius is 6370 km. Find the grid distance in meters in the netCDF file, and use the value for WRF's `namelist.input` file.

j. Using Digital Filter Initialization

Digital filter initialization (DFI) is a new option in V3. It is a way to remove initial model imbalance as, for example, measured by the surface pressure tendency. This might be important when one is interested in the 0 – 6 hour simulation/forecast. It runs a digital filter during a short model integration, backward and forward, and then starts the forecast. In WRF implementation, this is all done in a single job. With the V3.3 release, DFI can be used for multiple domains with concurrent nesting, with feedback disabled.

There is no special requirement for data preparation.

Start with the namelist template `namelist.input.dfi.` This namelist file contains an extra namelist record for DFI: `&dfi_control`. Edit it to match your case configuration. For a typical application, the following options are used:

`dfi_opt` = 3 (Note: if doing a restart, this must be changed to 0)
`dfi_nfilter` = 7 (filter option: Dolph)
`dfi_cutoff_seconds` = 3600 (should not be longer than the filter window)
For time specification, it typically needs to integrate backward for 0.5 to 1 hour, and integrate forward for half of the time.

If option `dfi_write_filtered_input` is set to true, a filtered wrfinput file, `wrfinput_initialized_d01`, will be produced when you run wrf.

In Version 3.2, a constant boundary condition option is introduced for DFI. To use it, set `constant_bc = 1` in `&bdy_control`

If a different time step is used for DFI, one may use `time_step_dfi` to set it.

k. Using sst_update option

The WRF model physics does not predict sea-surface temperature, vegetation fraction, albedo and sea ice. For long simulations, the model provides an alternative to read-in the time-varying data and update these fields. In order to use this option, one must have access to time-varying SST and sea ice fields. Twelve monthly values of vegetation fraction and albedo are available from the `geogrid` program. Once these fields are processed via WPS, one may activate the following options in the namelist record `&time_control` before running the program `real.exe` and `wrf.exe`:

`io_form_auxinput4` = 2
`auxinput4_inname` = "wrflowinp_d<domain>" (created by `real.exe`)
`auxinput4_interval` = 360, 360, 360,

and in *&physics*

`sst_update` = 1

l. Using Adaptive Time Stepping

Adaptive time stepping is a way to maximize the time step that the model can use while keeping the model numerically stable. The model time step is adjusted based on the domain-wide horizontal and vertical stability criterion. The following set of values would typically work well.

`use_adaptive_time_step` = .true.
`step_to_output_time` = .true. (but nested domains may still be writing output at the desired time. Try to use `adjust_output_times` = .true. to make up for this.)
`target_cfl` = 1.2, 1.2, 1.2,
`max_step_increase_pct` = 5, 51, 51, (a large percentage value for the nest allows the time step for the nest to have more freedom to adjust)
`starting_time_step` = the actual value or -1 (which means 6*DX at start time)
`max_time_step` : use fixed values for all domains, e.g. 8*DX
`min_time_step` : use fixed values for all domains, e.g. 4*DX

`adaptation_domain`: which domain is driving the adaptive time step

Also see the description of these options in the list of namelist on page 5-43.

m. Option to stochastically perturb forecast

Since Version 3.3, WRF has an option to stochastically perturb forecasts via a stochastic kinetic-energy backscatter scheme (SKEB, Shutts, 2005, QJRMS). The scheme introduces temporally and spatially correlated perturbations to the rotational wind components and potential temperature modulated by the total dissipation rate. In the current version, SKEBS 1.0, a spatially and temporally constant dissipation rate is assumed, but future developments will include a flow-dependent dissipation rate. There are several options for the vertical structure of the random pattern generator: barotropic and random phase. Details of the scheme are available in Berner et. al, 2011 (MWR, in press).

This scheme is controlled via the following physics namelist parameters, for each domain separately:

`stoch_force_opt` = 0, 0, 0 : No stochastic parameterization
= 1, 1, 1 : use SKEB scheme
`stoch_vertstruc_opt` = 0, 0, 0 : Constant vertical structure of random pattern generator
= 1, 1, 1 : Random phase vertical structure random pattern generator
`tot_backscat_psi` = Controls amplitude of rotational wind perturbations
(default value is 1.0E-5)
`tot_backscat_t` = Controls amplitude of potential temperature perturbations
(default value is 1.0E-6)

n. Run-Time IO

With the release of WRF version 3.2, IO decisions may now be updated as a run-time option. Previously, any modification to the IO (such as which variable is associated with which stream) was handled via the Registry, and changes to the Registry always necessitate a cycle of `clean -a`, `configure`, and `compile`. This compile-time mechanism is still available and it is how most of the WRF IO is defined. However, should a user wish to add (or remove) variables from various streams, that capability is available as an option.

First, the user lets the WRF model know where the information for the run-time modifications to the IO is located. This is a text file (`my_file_d01.txt`), one for each domain, defined in the `namelist.input` file, located in the `time_control` namelist record.

```
&time_control
iofields_filename = "my_file_d01.txt", "my_file_d02.txt"
ignore_iofields_warning = .true.,
/
```

The contents of the text file associates a stream ID (0 is the default history and input) with a variable, and whether the field is to be added or removed. The state variables must already be defined in the Registry file. Following are a few examples:

```
-:h:0:RAINC,RAINNC
```
would remove the fields RAINC and RAINNC from the standard history file.

`+:h:7:RAINC,RAINNC`
would add the fields RAINC and RAINNC to an output stream #7.

The available options are:

`+` or `-`, add or remove a variable
`0-24`, integer, which stream
`i` or `h`, input or history
field name in the Registry – this is the first string in quotes. Note: do not include any spaces in between field names.

It is not necessary to remove fields from one stream to insert them in another. It is OK to have the same field in multiple streams.

The second namelist variable, `ignore_iofields_warning`, tells the program what to do if it encounters an error in these user-specified files. The default value, .TRUE., is to print a warning message but continue the run. If set to .FALSE., the program will abort if there are errors in these user-specified files.

Note that any field that can be part of the optional IO (either the input or output streams) must already be declared as a state variable in the Registry. Care needs to be taken when specifying the names of the variables that are selected for the run-time IO. The "name" of the variable to use in the text file (defined in the namelist.input file) is the quoted string from the Registry file. Most of the WRF variables have the same string for the name of the variable used inside the WRF source code (column 3 in the Registry file, non-quoted, and not the string to use) and the name of the variable that appears in the netCDF file (column 9 in the Registry file, quoted, and that is the string to use).

o. Output Time Series

There is an option to output time series from a model run. To activate the option, a file called "`tslist`" must be present in the WRF run directory. The `tslist` file contains a list of locations defined by their latitude and longitude along with a short description and an abbreviation for each location. A sample file looks something like this:

```
#-----------------------------------------------#
# 24 characters for name | pfx |  LAT  |   LON  |
#-----------------------------------------------#
Cape Hallett             hallt -72.330  170.250
McMurdo Station          mcm   -77.851  166.713
```

The first three lines in the file are regarded as header information, and are ignored. Given a `tslist` file, for each location inside a model domain (either coarse or nested) a file containing time series variables at each model time step will be written with the name `pfx.d<domain>.TS`, where pfx is the specified prefix for the location in the tslist file. The maximum number of time series locations is controlled by the namelist variable `max_ts_locs` in the namelist record `&domains`. The default value is 5. The time series output contains selected variables at the surface, including 2-m temperature, vapor mixing ratio, 10-m wind components, u and v, rotated to the earth coordinate, etc.. More information for time series output can be found in `WRFV3/run/README.tslist`.

p. Using IO Quilting

This option allows a few processors to be set aside to be responsible for output only. It can be useful and performance-friendly if the domain size is large, and/or the time taken to write an output time is becoming

significant when compared to the time taken to integrate the model in between the output times. There are two variables for setting the option:

`nio_tasks_per_group`: How many processors to use per IO group for IO quilting. Typically 1 or 2 processors should be sufficient for this purpose.
`nio_groups`: How many IO groups for IO. Default is 1.

Examples of namelist for various applications

A few physics options sets (plus model top and the number of vertical levels) are provided here for reference. They may provide a good starting point for testing the model in your application. Also note that other factors will affect the outcome; for example, the domain setup, the distributions of vertical model levels, and input data.

a. 1 – 4 km grid distances, convection-permitting runs for a 1- 3 day run (as used for the NCAR spring real-time convection forecast over the US):

```
mp_physics                          = 8,
ra_lw_physics                       = 1,
ra_sw_physics                       = 2,
radt                                = 10,
sf_sfclay_physics                   = 2,
sf_surface_physics                  = 2,
bl_pbl_physics                      = 2,
bldt                                = 0,
cu_physics                          = 0,

ptop_requested                      = 5000,
e_vert                              = 35,
```

b. 20 – 30 km grid distances, 1- 3 day runs (e.g., NCAR daily real-time runs over the US):

```
mp_physics                          = 4,
ra_lw_physics                       = 1,
ra_sw_physics                       = 2,
radt                                = 10,
sf_sfclay_physics                   = 1,
sf_surface_physics                  = 2,
bl_pbl_physics                      = 1,
bldt                                = 0,
cu_physics                          = 1,
cudt                                = 5,

ptop_requested                      = 5000,
e_vert                              = 30,
```

c. Cold region 15 – 45 km grid sizes (e.g. used in NCAR's Antarctic Mesoscale Prediction System):

```
mp_physics                          = 4,
ra_lw_physics                       = 1,
ra_sw_physics                       = 2,
radt                                = 10,
```

```
sf_sfclay_physics                   = 2,
sf_surface_physics                  = 2,
bl_pbl_physics                      = 2,
bldt                                = 0,
cu_physics                          = 1,
cudt                                = 5,
fractional_seaice                   = 1,
seaice_threshold                    = 0.0,

ptop_requested                      = 1000,
e_vert                              = 44,
```

d. Hurricane applications (e.g. 12, 4 and 1.33 km nesting used by NCAR's real-time hurricane runs in 2010):

```
mp_physics                          = 8,
ra_lw_physics                       = 1,
ra_sw_physics                       = 2,
radt                                = 10,
sf_sfclay_physics                   = 1,
sf_surface_physics                  = 1,
bl_pbl_physics                      = 1,
bldt                                = 0,
cu_physics                          = 1,  (only on 12 km grid)
cudt                                = 5,
isftcflx                            = 2,

ptop_requested                      = 2000,
e_vert                              = 36,
```

e. Regional climate case at 10 – 30 km grid sizes (e.g. used in NCAR's regional climate runs):

```
mp_physics                          = 6,
ra_lw_physics                       = 3,
ra_sw_physics                       = 3,
radt                                = 30,
sf_sfclay_physics                   = 1,
sf_surface_physics                  = 2,
bl_pbl_physics                      = 1,
bldt                                = 0,
cu_physics                          = 1,
cudt                                = 5,
sst_update                          = 1,
tmn_update                          = 1,
sst_skin                            = 1,
bucket_mm                           = 100.0,
bucket_J                            = 1.e9,
ptop_requested                      = 1000,
e_vert                              = 51,

spec_bdy_width                      = 10,
spec_zone                           = 1,
relax_zone                          = 9,
```

```
spec_exp                            = 0.33,
```

Check Output

Once a model run is completed, it is good practice to check a couple of things quickly.

If you have run the model on multiple processors using MPI, you should have a number of `rsl.out.*` and `rsl.error.*` files. Type '`tail rsl.out.0000`' to see if you get '`SUCCESS COMPLETE WRF`'. This is a good indication that the model has run successfully.

The namelist options are written to a separate file: `namelist.output`.

Check the output times written to the `wrfout*` file by using the netCDF command:

```
ncdump -v Times wrfout_d01_yyyy-mm-dd_hh:00:00
```

Take a look at either the `rsl.out.0000` file or other standard-out files. This file logs the times taken to compute for one model time step, and to write one history and restart output file:

```
Timing for main: time 2006-01-21_23:55:00 on domain  2:    4.91110 elapsed seconds.
Timing for main: time 2006-01-21_23:56:00 on domain  2:    4.73350 elapsed seconds.
Timing for main: time 2006-01-21_23:57:00 on domain  2:    4.72360 elapsed seconds.
Timing for main: time 2006-01-21_23:57:00 on domain  1:   19.55880 elapsed seconds.
```

and

```
Timing for Writing wrfout_d02_2006-01-22_00:00:00 for domain 2: 1.17970 elapsed seconds.
Timing for main: time 2006-01-22_00:00:00 on domain 1: 27.66230 elapsed seconds.
Timing for Writing wrfout_d01_2006-01-22_00:00:00 for domain 1: 0.60250 elapsed seconds.
```

If the model did not run to completion, take a look at these standard output/error files too. If the model has become numerically unstable, it may have violated the CFL criterion (for numerical stability). Check whether this is true by typing the following:

`grep cfl rsl.error.*` or `grep cfl wrf.out`

you might see something like these:

```
5 points exceeded cfl=2 in domain            1 at time   4.200000
  MAX AT i,j,k:          123          48           3 cfl,w,d(eta)= 4.165821
21 points exceeded cfl=2 in domain            1 at time   4.200000
  MAX AT i,j,k:          123          49           4 cfl,w,d(eta)= 10.66290
```

When this happens, consider using the namelist option w_damping, and/or reducing the time step.

Trouble Shooting

If the model aborts very quickly, it is likely that either the computer memory is not large enough to run the specific configuration, or the input data have some serious problem. For the first problem, try to type '`unlimit`' or '`ulimit -s unlimited`' to see if more memory and/or stack size can be obtained.

For OpenMP (smpar-compiled code), the stack size needs to be set large, but not unlimited. Unlimited stack size may crash the computer.

To check if the input data is the problem, use ncview or another netCDF file browser.

Another frequent error seen is '`module_configure: initial_config: error reading namelist`'. This is an error message from the model complaining about errors and typos in the `namelist.input` file. Edit the `namelist.input` file with caution. If unsure, always start with an available template. A namelist record where the namelist read error occurs is provided in the V3 error message, and it should help with identifying the error.

Physics and Dynamics Options

Physics Options

WRF offers multiple physics options that can be combined in any way. The options typically range from simple and efficient, to sophisticated and more computationally costly, and from newly developed schemes, to well-tried schemes such as those in current operational models.

The choices vary with each major WRF release, but here we will outline those available in WRF Version 3.

1. Microphysics (mp_physics)

a. Kessler scheme: A warm-rain (i.e. no ice) scheme used commonly in idealized cloud modeling studies (*mp_physics* = 1).

b. Lin et al. scheme: A sophisticated scheme that has ice, snow and graupel processes, suitable for real-data high-resolution simulations (2).

c. WRF Single-Moment 3-class scheme: A simple, efficient scheme with ice and snow processes suitable for mesoscale grid sizes (3).

d. WRF Single-Moment 5-class scheme: A slightly more sophisticated version of (c) that allows for mixed-phase processes and super-cooled water (4).

e. Eta microphysics: The operational microphysics in NCEP models. A simple efficient scheme with diagnostic mixed-phase processes. For fine resolutions (< 5km) use option (5) and for coarse resolutions use option (95).

f. WRF Single-Moment 6-class scheme: A scheme with ice, snow and graupel processes suitable for high-resolution simulations (6).

g. Goddard microphysics scheme. A scheme with ice, snow and graupel processes suitable for high-resolution simulations (7). New in Version 3.0.

h. New Thompson et al. scheme: A new scheme with ice, snow and graupel processes suitable for high-resolution simulations (8). This adds rain number concentration and updates the scheme from the one in Version 3.0. New in Version 3.1.

i. Milbrandt-Yau Double-Moment 7-class scheme (9). This scheme includes separate categories for hail and graupel with double-moment cloud, rain, ice, snow, graupel and hail. New in Version 3.2.

j. Morrison double-moment scheme (10). Double-moment ice, snow, rain and graupel for cloud-resolving simulations. New in Version 3.0.

k. WRF Double-Moment 5-class scheme (14). This scheme has double-moment rain. Cloud and CCN for warm processes, but is otherwise like WSM5. New in Version 3.1.

l. WRF Double-Moment 6-class scheme (16). This scheme has double-moment rain. Cloud and CCN for warm processes, but is otherwise like WSM6. New in Version 3.1.

m. Stony Brook University (Y. Lin) scheme (13). This is a 5-class scheme with riming intensity predicted to account for mixed-phase processes. New in Version 3.3.

n. NSSL 2-moment scheme (17, 18). This is a two-moment scheme for cloud droplets, rain drops, ice crystals, snow, graupel, and hail. It also predicts average graupel particle density, which allows graupel to span the range from frozen drops to low-density graupel. There is an additional option to predict cloud condensation nuclei (CCN, option 18) concentration (intended for idealized simulations). The scheme is intended for cloud-resolving simulations (dx <= 2km) in research applications. New in Version 3.4.

2.1 Longwave Radiation (ra_lw_physics)

a. RRTM scheme: Rapid Radiative Transfer Model. An accurate scheme using look-up tables for efficiency. Accounts for multiple bands, trace gases, and microphysics species (*ra_lw_physics* = 1).

b. GFDL scheme: Eta operational radiation scheme. An older multi-band scheme with carbon dioxide, ozone and microphysics effects (99).

c. CAM scheme: from the CAM 3 climate model used in CCSM. Allows for aerosols and trace gases (3).

d. RRTMG scheme. A new version of RRTM added in Version 3.1 (4). It includes the MCICA method of random cloud overlap.

e. New Goddard scheme (5). Efficient, multiple bands, ozone from climatology. New in Version 3.3.

f. Fu-Liou-Gu scheme (7). multiple bands, cloud and cloud fraction effects, ozone profile from climatology. New in Version 3.4.

2.2 Shortwave Radiation (ra_sw_physics)

a. Dudhia scheme: Simple downward integration allowing efficiently for clouds and clear-sky absorption and scattering (*ra_sw_physics* = 1).

b. Goddard shortwave: Two-stream multi-band scheme with ozone from climatology and cloud effects (2).

c. GFDL shortwave: Eta operational scheme. Two-stream multi-band scheme with ozone from climatology and cloud effects (99).

d. CAM scheme: from the CAM 3 climate model used in CCSM. Allows for aerosols and trace gases (3).

e. RRTMG shortwave. A new shortwave scheme with the MCICA method of random cloud overlap (4). New in Version 3.1.

f. New Goddard scheme (5). Efficient, multiple bands, ozone from climatology. New in Version 3.3.

g. Fu-Liou-Gu scheme (7). multiple bands, cloud and cloud fraction effects, ozone profile from climatology, can allow for aerosols. New in Version 3.4.

h. Held-Suarez relaxation. A temperature relaxation scheme designed for idealized tests only (31).

i. Slope and shading effects. *slope_rad* = 1 modifies surface solar radiation flux according to terrain slope. *topo_shad* = 1 allows for shadowing of neighboring grid cells. Use only with high-resolution runs with grid size less than a few kilometers. Since Version 3.2, these are available for all shortwave options.

j. *swrad_scat*: scattering turning parameter for *ra_sw_physics* = 1. Default value is 1, which is equivalent to 1.e-5 m^2/kg. When the value is greater than 1, it increases the scattering.

3.1 Surface Layer (sf_sfclay_physics)

a. MM5 similarity: Based on Monin-Obukhov with Carslon-Boland viscous sub-layer and standard similarity functions from look-up tables (*sf_sfclay_physics* = 1).

b. Eta similarity: Used in Eta model. Based on Monin-Obukhov with Zilitinkevich thermal roughness length and standard similarity functions from look-up tables (2).

c. Pleim-Xiu surface layer. (7). New in Version 3.0.

d. QNSE surface layer. Quasi-Normal Scale Elimination PBL scheme's surface layer option (4). New in Version 3.1.

e. MYNN surface layer. Nakanishi and Niino PBL's surface layer scheme (5). New in Version 3.1.

f. TEMF surface layer. Total Energy – Mass Flux surface layer scheme. New in Version 3.3.

g. Revised MM5 surface layer scheme (11): Remove limits and use updated stability functions. New in Version 3.4. (Jimenez et al. MWR 2012).

h. *iz0tlnd* = 1 (for *sf_sfclay_physics* = 1 or 2), Chen-Zhang thermal roughness length over land, which depends on vegetation height, 0 = original thermal roughness length in each sfclay option. New in Vserion 3.2.

3.2 Land Surface (sf_surface_physics)

a. 5-layer thermal diffusion: Soil temperature only scheme, using five layers (*sf_surface_physics* = 1).

b. Noah Land Surface Model: Unified NCEP/NCAR/AFWA scheme with soil temperature and moisture in four layers, fractional snow cover and frozen soil physics. New modifications are added in Version 3.1 to better represent processes over ice sheets and snow covered area.

c. RUC Land Surface Model: RUC operational scheme with soil temperature and moisture in six layers, multi-layer snow and frozen soil physics (3).

d. Pleim-Xiu Land Surface Model. Two-layer scheme with vegetation and sub-grid tiling (7). New in Version 3.0.

f. Noah-MP (multi-physics) Land Surface Model: uses multiple options for key land-atmosphere interaction processes. Noah-MP contains a separate vegetation canopy defined by a canopy top and bottom with leaf physical and radiometric properties used in a two-stream canopy radiation transfer scheme that includes shading effects. Noah-MP contains a multi-layer snow pack with liquid water storage and melt/refreeze capability and a snow-interception model describing loading/unloading, melt/refreeze, and sublimation of the canopy-intercepted snow. Multiple options are available for surface water infiltration and runoff, and groundwater transfer and storage including water table depth to an unconfined aquifer. Horizontal and vertical vegetation density can be prescribed or predicted using prognostic photosynthesis and dynamic vegetation models that allocate carbon to vegetation (leaf, stem, wood and root) and soil carbon pools (fast and slow). New in Version 3.4. (Niu et al. 2011)

g. SSiB Land Surface Model: This is the third generation of the Simplified Simple Biosphere Model (Xue et al. 1991; Sun and Xue, 2001). SSiB is developed for land/atmosphere interaction studies in the climate model. The aerodynamic resistance values in SSiB are determined in terms of vegetation properties, ground conditions and bulk Richardson number according to the modified Monin–Obukhov similarity theory. SSiB-3 includes three snow layers to realistically simulate snow processes, including destructive metamorphism, densification process due to snow load, and snow melting, which substantially enhances the model's ability for the cold season study. To use this option, *ra_lw_physics* and *ra_sw_physics* should be set to either 1, 3, or 4. The second full model level should be set to no larger than 0.982 so that the height of that level is higher than vegetation height. New in Version 3.4.

h. Fractional sea-ice (*fractional_seaice* = 1). Treat sea-ice as fractional field. Require fractional sea-ice as input data. Data sources may include those from GFS or the National Snow and Ice Data Center (http://nsidc.org/data/seaice/index.html). Use XICE for Vtable entry instead of SEAICE. This option works with *sf_sfclay_physics* = 1, 2, and *sf_surface_physics* = 2, 3 in the present release. New in Version 3.1.

3.3 Urban Surface (sf_urban_physics – replacing old switch ucmcall)

a. Urban canopy model (1): 3-category UCM option with surface effects for roofs, walls, and streets.

b. BEP (2). Building Environment Parameterization: Multi-layer urban canopy model that allows for buildings higher than the lowest model levels. Only works with Noah LSM and Boulac and MYJ PBL options. New in Version 3.1.

c. BEM (3). Building Energy Model. Adds to BEP, building energy budget with heating and cooling systems. Works with same options as BEP. New in Version 3.2.

4. Planetary Boundary layer (bl_pbl_physics)

a. Yonsei University scheme: Non-local-K scheme with explicit entrainment layer and parabolic K profile in unstable mixed layer (*bl_pbl_physics* = 1).

b. Mellor-Yamada-Janjic scheme: Eta operational scheme. One-dimensional prognostic turbulent kinetic energy scheme with local vertical mixing (2).

c. MRF scheme: Older version of (a) with implicit treatment of entrainment layer as part of non-local-K mixed layer (99).

d. ACM2 PBL: Asymmetric Convective Model with non-local upward mixing and local downward mixing (7). New in Version 3.0.

e. Quasi-Normal Scale Elimination PBL (4). A TKE-prediction option that uses a new theory for stably stratified regions (Available since 3.1). Daytime part uses eddy diffusivity mass-flux method with shallow convection (*mfshconv* = 1) which is added in Version 3.4.

f. Mellor-Yamada Nakanishi and Niino Level 2.5 PBL (5). Predicts sub-grid TKE terms. New in Version 3.1.

g. Mellor-Yamada Nakanishi and Niino Level 3 PBL (6). Predicts TKE and other second-moment terms. New in Version 3.1.

h. BouLac PBL (8): Bougeault-Lacarrère PBL. A TKE-prediction option. New in Version 3.1. Designed for use with BEP urban model.

i. UW (Bretherton and Park) scheme (9). TKE scheme from CESM climate model. New in Version 3.3.

j. Total Energy - Mass Flux (TEMF) scheme (10). Sub-grid total energy prognostic variable, plus mass-flux type shallow convection. New in Version 3.3.

k. LES PBL: A large-eddy-simulation (LES) boundary layer is available in Version 3. For this, *bl_pbl_physic* = 0, *isfflx* = 1, and *sf_sfclay_physics* and *sf_surface_physics* are selected. This uses diffusion for vertical mixing and must use *diff_opt* = 2, and *km_opt* = 2 or 3, see below. Alternative idealized ways of running the LESPBL are chosen with *isfflx* = 0 or 2. New in Version 3.0.

l. *topo_wind* = 1: Topographic correction for surface winds to represent extra drag from sub-grid topography and enhanced flow at hill tops (Jimenez and Dudhia, JAMC 2012). Works with YSU PBL only. New in Version 3.4.

5. Cumulus Parameterization (cu_physics)

a. Kain-Fritsch scheme: Deep and shallow convection sub-grid scheme using a mass flux approach with downdrafts and CAPE removal time scale (*cu_physics* = 1).

- kfeta_trigger = 1 – default trigger; = 2 – moisture-advection modulated trigger function [based on Ma and Tan (2009, Atmospheric Research)]. May improve results in subtropical regions when large-scale forcing is weak.

b. Betts-Miller-Janjic scheme. Operational Eta scheme. Column moist adjustment scheme relaxing towards a well-mixed profile (2).

c. Grell-Devenyi (GD) ensemble scheme: Multi-closure, multi-parameter, ensemble method with typically 144 sub-grid members (3).

d. Simplified Arakawa-Schubert (4). Simple mass-flux scheme with quasi-equilibrium closure with shallow mixing scheme (and momentum transport in NMM only). Adapted for ARW in Version 3.3.

e. Grell 3D is an improved version of the GD scheme that may also be used on high resolution (in addition to coarser resolutions) if subsidence spreading (option cugd_avedx) is turned on (5). New in Version 3.0.

f. Tiedtke scheme (U. of Hawaii version) (6). Mass-flux type scheme with CAPE-removal time scale, shallow component and momentum transport. New in Version 3.3.

g. Zhang-McFarlane scheme (7). Mass-flux CAPE-removal type deep convection from CESM climate model with momentum transport. New in Version 3.3.

h. New Simplified Arakawa-Schubert (14). New mass-flux scheme with deep and shallow components and momentum transport. New in Version 3.3.

i. New Simplified Arakawa-Schubert (84, HWRF version). New mass-flux scheme with deep and shallow components and momentum transport. New in Version 3.4.

j. Old Kain-Fritsch scheme: Deep convection scheme using a mass flux approach with downdrafts and CAPE removal time scale (99).

6. Shallow convection option (shcu_physics)

a. *ishallow* = 1, shallow convection option on. Works together with Grell 3D scheme (cu_physics = 5) – will move to shcu_physics category in the future

b. UW (Bretherton and Park) scheme (2). Shallow cumulus option from CESM climate model with momentum transport. New in Version 3.3.

7. Other physics options

a. Options to use for tropical storm and hurricane applications:

- *omlcall* = 1: Simple ocean mixed layer model (1): 1-D ocean mixed layer model following that of Pollard, Rhines and Thompson (1972). Two other namelist options are available to specify the initial mixed layer depth (although one may ingest real mixed layer depth data) (*oml_hml0*) and temperature lapse rate below the mixed layer (*oml_gamma*). Since V3.2, this option works with all *sf_surface_physics* options.

- *isftcflx*: Modify surface bulk drag (Donelan) and enthalpy coefficients to be more in line with recent research results of those for tropical storms and hurricanes. This option also includes dissipative heating term in heat flux. It is only available for *sf_sfclay_physics* = 1. There are two options for computing enthalpy coefficients: *isftcflx* = 1: constant Z_{0q} (since V3.2) for heat and moisture; *isftcflx* = 2 Garratt formulation, slightly different forms for heat and moisture.

b. Other options for long simulations (new in Version 3.1):

- *tmn_update*: update deep soil temperature (1).

- *sst_skin*: calculate skin SST based on Zeng and Beljaars (2005) (1)

- *bucket_mm*: bucket reset value for water equivalent precipitation accumulations (value in mm, -1 = inactive).

- *bucket_J*: bucket reset value for energy accumulations (value in Joules, -1 = inactive). Only works with CAM and RRTMG radiation (*ra_lw_physics* = 3 and 4 and *ra_sw_physics* = 3 and 4) options.

- To drive WRF model with climate data that does not have leap year, there is a compile option to do that. Edit `configure.wrf` and
add -DNO_LEAP_CALENDAR to the macro ARCH_LOCAL.

c. *usemonalb*: When set to .true., it uses monthly albedo fields from geogrid, instead of table values

d. *no_mp_heating*: When set to 1, it turns off latent heating from microphysics. When using this option, *cu_physics* should be set to 0.

e. *gwd_opt*: Gravity wave drag option. Can be activated when grid size is greater than 10 km. May be beneficial for simulations longer than 5 days and over a large domain with mountain ranges. New in Version 3.1.

f. *windturbines_spec* (a character string): Wind turbine drag parameterization scheme. It represents sub-grid effects of specified turbines on wind and TKE fields. When set to “none” (default value), the scheme is off. When set to “ideal”, the idealized specification for turbine’s geometry and characteristics are set by namelist variables td_*. When set to a file name (which exists in the run directory), the physical charateristics of the wind farm is described in the file. See README.windturbine in WRFV3/ directory for more detail. New in

Version 3.3, and in this version it only works with 2.5 level MYNN PBL option (*bl_pbl_physics*=5).

8. Physics sensitivity options

a. *no_mp_heating*: When set to 1, it turns off latent heating from microphysics. When using this option, *cu_physics* should be set to 0.
b. *icloud*: When set to 0, it turns off cloud effect on optical depth in shortwave radiation options 1, 4 and longwave radiation option 1, 4.
c. *isfflx*: When set to 0, it turns off both sensible and latent heat fluxes from the surface. This option works for *sf_sfclay_physics* = 1, 5, 7, 11.
d. *ifsnow*: When set to 0, it turns off snow effect in *sf_surface_physics* = 1.

Diffusion and Damping Options

Diffusion in WRF is categorized under two parameters: the diffusion option and the K option. The diffusion option selects how the derivatives used in diffusion are calculated, and the K option selects how the K coefficients are calculated. Note that when a PBL option is selected, vertical diffusion is done by the PBL scheme, and not by the diffusion scheme. In Version 3, vertical diffusion is also linked to the surface fluxes.

1.1 Diffusion Option (diff_opt)

a. Simple diffusion: Gradients are simply taken along coordinate surfaces (*diff_opt* = 1).

b. Full diffusion: Gradients use full metric terms to more accurately compute horizontal gradients in sloped coordinates (*diff_opt* = 2).

1.2 K Option (km_opt)

Note that when using a PBL scheme, only options (a) and (d) below make sense, because (b) and (c) are designed for 3d diffusion.

a. Constant: K is specified by namelist values for horizontal and vertical diffusion (*km_opt* = 1).

b. 3d TKE: A prognostic equation for turbulent kinetic energy is used, and K is based on TKE (*km_opt* = 2).

c. 3d Deformation: K is diagnosed from 3d deformation and stability following a Smagorinsky approach (*km_opt* = 3).

d. 2d Deformation: K for horizontal diffusion is diagnosed from just horizontal deformation. The vertical diffusion is assumed to be done by the PBL scheme (*km_opt* = 4).

1.3 6th Order Horizontal Diffusion (diff_6th_opt)

6^{th}-order horizontal hyper diffusion (del^6) on all variables to act as a selective short-wave numerical noise filter. Can be used in conjunction with *diff_opt*. = 1: simple; = 2: positive definite. Option 2 is recommended.

1.4 Nonlinear Backscatter Anisotropic (NBA) (sfs_opt)

Sub-grid turbulent stress option for momentum in LES applications. New in Version 3.2. *sfs_opt* = 1 diagnostic sub-grid stress to be used with *diff_opt* = 2 and *km_opt* = 2 or 3. *sfs_opt* = TKE sub-grid stress to be used with *diff_opt* = 2 and *km_opt* = 2.

2. Damping Options

These are independently activated choices.

a. Upper Damping: Either a layer of increased diffusion (*damp_opt* =1) or a Rayleigh relaxation layer (2) or an implicit gravity-wave damping layer (3, new in Version 3.0), can be added near the model top to control reflection from the upper boundary.

b. Vertical velocity damping (*w_damping*): For operational robustness, vertical motion can be damped to

prevent the model from becoming unstable with locally large vertical velocities. This only affects strong updraft cores, so has very little impact on results otherwise.

c. Divergence Damping (*sm_div*): Controls horizontally-propagating sound waves.

d. External Mode Damping (*em_div*): Controls upper-surface (external) waves.

e. Time Off-centering (*epssm*): Controls vertically-propagating sound waves.

Advection Options

a. Horizontal advection orders for momentum (*h_mom_adv_order*) and scalar (*h_sca_adv_order*) can be $2^{nd}$to 6^{th}, with 5^{th} order being the recommended one.

b. Vertical advection orders for momentum (*v_mom_adv_order*) and scalar (v_sca_adv_order) can be $2^{nd}$and 6th, with 3^{rd} order being the recommended one.

c. Monotonic transport (option 2, new in Version 3.1) and positive-definite advection option (option 1) can be applied to moisture (*moist_adv_opt*), scalar (*scalar_adv_opt*), chemistry variables (*chem_adv_opt*) and tke (*tke_adv_opt*). Option 1 replaces *pd_moist* = .true. etc. in previous versions.

d. WENO (weighted essentially non-oscillatory) (option 3 for 5^{th} order WENO; option 4 for 5^{th} order WENO with positive definite limiter): for moisture (*moist_adv_opt*), scalar (*scalar_adv_opt*), chemistry variables (*chem._adv_opt*) and TKE (*tke_adv_opt*). For momentum, *momentum_adv_opt* = 3.

Some notes about using monotonic and positive-definite advection options:

The positive-definite and monotonic options are available for moisture, scalars, chemical scalers and TKE in the ARW solver. Both the monotonic and positive-definite transport options conserve scalar mass locally and globally and are consistent with the ARW mass conservation equation. We recommend using the positive-definite option for moisture variables on all real-data simulations. The monotonic option may be beneficial in chemistry applications and for moisture and scalars in some instances.

When using these options there are certain aspects of the ARW integration scheme that should be considered in the simulation configuration.

(1) The integration sequence in ARW changes when the positive-definite or monotonic options are used. When the options are not activated, the timestep tendencies from the physics (excluding microphysics) are used to update the scalar mixing ratio at the same time as the transport (advection). The microphysics is computed, and moisture is updated, based on the transport+physics update. When the monotonic or positive definite options are activated, the scalar mixing ratio is first updated with the physics tendency, and the new updated values are used as the starting values for the transport scheme. The microphysics update occurs after the transport update using these latest values as its starting point. It is important to remember that for any scalars, the local and global conservation properties, positive definiteness and monotonicity depend upon each update possessing these properties.

(2) Some model filters may not be positive definite.

i. *diff_6th_opt* = 1 is not positive definite nor monotonic. Use *diff_6th_opt* = 2 if you need this diffusion option (*diff_6th_opt* = 2 is monotonic and positive-definite). We have encountered cases where the departures from monotonicity and positive-definiteness have been very noticeable.
ii. *diff_opt* = 1 and *km_opt* = 4 (a commonly-used real-data case mixing option) is not guaranteed to be positive-definite nor monotonic due to the variable eddy diffusivity, K. We have not observed significant departures from positive-definiteness or monotonicity when this filter is used with these transport options.
iii. The diffusion option that uses a user-specified constant eddy viscosity is positive definite and

monotonic.

iv. Other filter options that use variable eddy viscosity are not positive definite or monotonic.

(3) Most of the model physics are not monotonic nor should they be - they represent sources and sinks in the system. All should be positive definite, although we have not examined and tested all options for this property.

(4) The monotonic option adds significant smoothing to the transport in regions where it is active. You may want to consider turning off the other model filters for variables using monotonic transport (filters such as the second and sixth order horizontal filters). At present it is not possible to turn off the filters for the scalars but not for the dynamics using the namelist - one must manually comment out the calls in the solver.

Other Dynamics Options

a. The model can be run hydrostatically by setting the *non_hydrostatic* switch to .false.

b. The Coriolis term can be applied to wind perturbation (*pert_coriolis* = .true.) only (idealized only).

c. For *diff_opt* = 2 only, vertical diffusion may act on full fields (not just on perturbation from the 1D base profile (*mix_full_fields* = .true.; idealized only).

Lateral Boundary Condition Options

a. Periodic (*periodic_x* / *periodic_y*): for idealized cases.

b. Open (*open_xs, open_xe, open_ys, open_ye*): for idealized cases.

c. Symmetric (*symmetric_xs, symmetric_xe, symmetric_ys, symmetric_ye*): for idealized cases.

d. Specified (*specified*): for real-data cases. The first row and column are specified with external model values (*spec_zone* = 1, and it should not change). The rows and columns in *relax_zone* have values blended from an external model and WRF. The value of *relax_zone* may be changed, as long as *spec_bdy_width* = *spec_zone* + *relax_zone*. This can be used with *periodic_x* in tropical channel simulations.

spec_exp: exponential multiplier for the relaxation zone ramp, used with a *specified* boundary condition. 0. = linear ramp, default; 0.33 = ~3*dx exp decay factor. This may be useful for long simulations.

e. Nested (*nested*): for real and idealized cases.

Summary of PBL Physics Options

bl_pbl_physics	Scheme	Reference	Added
1	YSU	Hong, Noh and Dudhia (2006, MWR)	2004
2	MYJ	Janjic (1994, MWR)	2000
3	GFS	Hong and Pan (1996, MWR)	2005
4	QNSE	Sukoriansky, Galperin and Perov (2005, BLM)	2009
5	MYNN2	Nakanishi and Niino (2006, BLM)	2009
6	MYNN3	Nakanishi and Niino (2006, BLM)	2009
7	ACM2	Pleim (2007, JAMC	2008

8	BouLac	Bougeault and Lacarrere (1989, MWR)	2009
9	UW	Bretherton and Park (2009, JC)	2011
10	TEMF	Angevine, Jiang and Mauriten (2010, MWR)	2011
99	MRF	Hong and Pan (1996, MWR)	2000

bl_pbl_physics	**Scheme**	**Cores**	**sf_sfclay_physics**	**Prognostic variables**	**Diagnostic variables**	**Cloud mixing**
1	YSU	ARW/ NMM	1		exch_h	QC,QI
2	MYJ	ARW/ NMM	2	TKE_PBL	EL_MYJ, exch_h	QC,QI
3	GFS (hwrf)	NMM	3			QC,QI
4	QNSE	ARW/ NMM	4	TKE_PBL	EL_MYJ, exch_h, exch_m	QC,QI
5	MYNN2	ARW	1,2,5	QKE	Tsq, Qsq, Cov, exch_h, exch_m	QC
6	MYNN3	ARW	1,2,5	QKE, Tsq, Qsq, Cov	exch_h, exch_m	QC
7	ACM2	ARW	1,7			QC,QI
8	BouLac	ARW	1,2	TKE_PBL	EL_PBL, exch_h, exch_m, wu_tur, wv_tur, wt_tur, wq_tur	QC
9	UW	ARW	1,2	TKE_PBL	exch_h, exch_m	QC
10	TEMF	ARW	10	TE_TEMF	*_temf	QC, QI
99	MRF	ARW/ NMM	1			QC,QI

Summary of Microphysics Options

mp_physics	**Scheme**	**Reference**	**Added**
1	Kessler	Kessler (1969)	2000

2	Lin (Purdue)	Lin, Farley and Orville (1983, JCAM)	2000
3	WSM3	Hong, Dudhia and Chen (2004, MWR)	2004
4	WSM5	Hong, Dudhia and Chen (2004, MWR)	2004
5	Eta (Ferrier)	Rogers, Black, Ferrier, Lin, Parrish and DiMego (2001, web doc)	2000
6	WSM6	Hong and Lim (2006, JKMS)	2004
7	Goddard	Tao, Simpson and McCumber (1989, MWR)	2008
8	Thompson	Thompson, Field, Rasmussen and Hall (2008, MWR)	2009
9	Milbrandt 2-mom	Milbrandt and Yau (2005, JAS)	2010
10	Morrison 2-mom	Morrison, Thompson and Tatarskii (2009, MWR)	2008
13	SBU-YLin	Lin and Colle (2011, MWR)	2011
14	WDM5	Lim and Hong (2010, MWR)	2009
16	WDM6	Lim and Hong (2010, MWR)	2009
17	NSSL 2-mom	Mansell, Ziegler and Bruning (2010, JAS)	2012
18	NSSL 2-mom w/ CCN prediction	Mansell, Ziegler and Bruning (2010, JAS)	2012

mp_physics	**Scheme**	**Cores**	**Mass Variables**	**Number Variables**
1	Kessler	ARW	Qc Qr	
2	Lin (Purdue)	ARW (Chem)	Qc Qr Qi Qs Qg	
3	WSM3	ARW	Qc Qr	
4	WSM5	ARW/NMM	Qc Qr Qi Qs	
5	Eta (Ferrier)	ARW/NMM	Qc Qr Qs (Qt*)	
6	WSM6	ARW/NMM	Qc Qr Qi Qs Qg	
8	Thompson	ARW/NMM	Qc Qr Qi Qs Qg	Ni Nr
9	Milbrandt 2-mom	ARW	Qc Qr Qi Qs Qg Qh	Nc Nr Ni Ns Ng Nh
10	Morrison 2-mom	ARW (Chem)	Qc Qr Qi Qs Qg	Nr Ni Ns Ng
13	SBU-YLin	ARW	Qc Qr Qi Qs	
14	WDM5	ARW	Qc Qr Qi Qs	Nn** Nc Nr

16	WDM6	ARW	Qc Qr Qi Qs Qg	Nn** Nc Nr
17	NSSL 2-mom	ARW	Qc Qr Qi Qs Qg Qh	Nc Nr Ni Ns Ng Nh
18	NSSL 2-mom +CCN	ARW	Qc Qr Qi Qs Qg Qh	Nc Nr Ni Ns Ng Nh Nn

* Advects only total condensates ** Nn = CCN number

Summary of Cumulus Parameterization Options

cu_physics	Scheme	Reference	Added
1	Kain-Fritsch	Kain (2004, JAM)	2000
2	Betts-Miller-Janjic	Janjic (1994, MWR; 2000, JAS)	2002
3	Grell-Devenyi	Grell and Devenyi (2002, GRL)	2002
4	Simplied Arakawa-Schubert	Pan and Wu (1995), NMC Office Note 409	2005/ 2011
5	Grell-3	-	2008
6	Tiedtke	Tiedtke (1989, MWR), Zhang et al. (2011, MWR)	2011
7	Zhang-McFarlane	Zhang and McFarlane (1995, AO)	2011
14	New SAS	Han and Pan (2011, Wea. Forecasting)	2011
84	New SAS (HWRF)	Han and Pan (2011, Wea. Forecasting)	2012
99	Old Kain-Fritsch	Kain and Fritsch (1990, JAS; 1993, Meteo. Monogr.)	2000

cu_physics	Scheme	Cores	Moisture Tendencies	Momentum Tendencies	Shallow Convection
1	Kain-Fritsch	ARW / NMM	Qc Qr Qi Qs	no	yes
2	BMJ	ARW / NMM	-	no	yes
3	GD	ARW	Qc Qi	no	no
4	SAS	ARW / NMM	Qc Qi	yes (NMM)	yes (ARW)
5	G3	ARW	Qc Qi	no	yes
6	Tiedtke	ARW	Qc Qi	yes	yes
7	Zhang-McFarlane	ARW	Qc Qi	yes	no
14	NSAS	ARW	Qc Qr Qi Qs	yes	yes

99	old KF	ARW	Qc Qr Qi Qs	no	no

Description of Namelist Variables

The following is a description of the namelist variables. The variables that are a function of nests are indicated by *(max_dom)* following the variable. Also see the `Registry/Registry.EM` and `run/README.namelist` files in the WRFV3/ directory.

Variable Names	Value	Description
`&time_control`		Time control
`run_days`	1	run time in days
`run_hours`	0	run time in hours Note: if it is more than 1 day, one may use both run_days and run_hours or just run_hours. e.g. if the total run length is 36 hrs, you may set run_days = 1, and run_hours = 12, or run_days = 0, and run_hours 36
`run_minutes`	0	run time in minutes
`run_seconds`	0	run time in seconds
`start_year (max_dom)`	2001	four digit year of starting time
`start_month (max_dom)`	06	two digit month of starting time
`start_day (max_dom)`	11	two digit day of starting time
`start_hour (max_dom)`	12	two digit hour of starting time
`start_minute (max_dom)`	00	two digit minute of starting time
`start_second (max_dom)`	00	two digit second of starting time Note: the start time is used to name the first wrfout file. It also controls the start time for nest domains, and the time to restart
`end_year (max_dom)`	2001	four digit year of ending time
`end_month (max_dom)`	06	two digit month of ending time
`end_day (max_dom)`	12	two digit day of ending time
`end_hour (max_dom)`	12	two digit hour of ending time
`end_minute (max_dom)`	00	two digit minute of ending time
`end_second (max_dom)`	00	two digit second of ending time Note all end times also control when the nest domain integrations end. All start and end times are used by *real.exe*. One may use either run_days/run_hours etc. or end_year/month/day/hour etc. to control the length of model integration. But run_days/run_hours takes precedence over the end times. The program *real.exe* uses start and end times only.

interval_seconds	10800	time interval between incoming real data, which will be the interval between the lateral boundary condition file (for *real* only)
input_from_file (max_dom)	T (logical)	logical; whether the nested run will have input files for domains other than 1
fine_input_stream (max_dom)		selected fields from nest input
	0	all fields from nest input are used
	2	only nest input specified from input stream 2 (defined in the Registry) are used. In V3.2, this requires io_form_auxinput2 to be set
history_interval (max_dom)	60	history output file interval in minutes (integer only)
history_interval_d (max_dom)	1	history output file interval in days (integer); used as an alternative to history_interval
history_interval_h (max_dom)	1	history output file interval in hours (integer); used as an alternative to history_interval
history_interval_m (max_dom)	1	history output file interval in minutes (integer); used as an alternative to history_interval, and is equivalent to history_interval
history_interval_s (max_dom)	1	history output file interval in seconds (integer); used as an alternative to history_interval
frames_per_outfile (max_dom)	1	output times per history output file, used to split output files into smaller pieces
restart	F (logical)	whether this run is a restart run
restart_interval	1440	restart output file interval in minutes
override_restart_timers	F	T: reset history and restart output intervals from those in restart file
write_hist_at_0h_rst	F	T: allows one to write history file at the time of restart
reset_simulation_start	F	whether to overwrite the simulation_start_date with the forecast start time
cycling	F	whether this run is a cycling run (initialized from wrfout file)
auxinput1_inname	"met_em.d<domain> <date>"	input from WPS (this is the default)
auxinput4_inname	"wrflowinp_d<domain>"	input for lower bdy file, works with sst_update = 1
auxinput4_interval (max_dom)	360	file interval in minutes for lower boundary file

io_form_auxinput4	2	IO format for wrflowinp files, required for V3.2
io_form_history	2	2 = netCDF; 102 = split netCDF files one per processor (no supported post-processing software for split files)
	1	binary format (no supported post-processing software avail)
	4	PHDF5 format (no supported post-processing software avail)
	5	GRIB 1
	10	GRIB 2
	11	parallel netCDF
io_form_restart	2	2 = netCDF; 102 = split netCDF files one per processor (must restart with the same number of processors)
io_form_input	2	2 = netCDF
	102	allows the program `real.exe` to read in split `met_em*` files, and write split `wrfinput` files. No split file for `wrfbdy`.
io_form_boundary	2	netCDF format
io_form_auxinput4	2	IO format (netCDF) for wrflowinp
io_form_auxinput2	2	IO format (netCDF) for input stream 2 data
cycling	.false.	indicating if the run is using the `wrfout` file as the input file. In this case the Thompson initialization routine will not be called again (performance issue)
diag_print	0	getting some simple diagnostic fields
	1	domain averaged Dpsfc/Dt, Dmu/Dt will appear in stdout file
	2	in addition to those above, domain averaged rainfall, surface evaporation, and sensible and latent heat fluxes will be output
debug_level	0	50,100,200,300 values give increasing prints
auxhist2_outname	"rainfall_d<domain>"	file name for extra output; if not specified, auxhist2_d<domain>_<date> will be used. Also note that to write variables in output other than the history file requires a Registry.EM file change
auxhist2_interval (max_dom)	10	interval in minutes
io_form_auxhist2	2	output in netCDF
frames_per_auxhist2 (max_dom)		output times per output file
auxinput11_interval		designated for obs nudging input
auxinput11_end_h		designated for obs nudging input
nocolons	.false.	replace : with _ in output file names

write_input	t	write input-formatted data as output for 3DVAR application
inputout_interval	180	interval in minutes when writing input-formatted data
input_outname	"wrf_3dvar_input_ d<domain>_<date>"	Output file name from 3DVAR
inputout_begin_y	0	beginning year to write 3DVAR data
inputout_begin_d	0	beginning day to write 3DVAR data
inputout_begin_h	3	beginning hour to write 3DVAR data
Inputout_begin_m	0	beginning minute to write 3DVAR data
inputout_begin_s	0	beginning second to write 3DVAR data
inputout_end_y	0	ending year to write 3DVAR data
inputout_end_d	0	ending day to write 3DVAR data
inputout_end_h	12	ending hour to write 3DVAR data
inputout_end_m	0	ending minute to write 3DVAR data
inputout_end_s	0	ending second to write 3DVAR data. The above example shows that the input-formatted data are output starting from hour 3 to hour 12 in a 180-min interval.
all_ic_times	1	output wrfinput file for all time periods
output_diagnostics	0	1: 36 surface diagnostic arrays (max/min /mean/std) in the time interval specified. The output goes to auxiliary history output stream 3 with default file name 'wrfxtrm_d<domain>_<date>'. Also set io_form_auxhist3 = 2 auxhist3_interval = 1440, 1440, frames_per_auxhist3 = 1000,1000,
iofields_filename (max_dom)	"my_iofields_list"	a text file with a list of IO fields. e.g. +:h:21:rainc, rainnc, rthcuten
ignore_iofields _warning	.true.	what to do when encountering an error in the user-specified files. .false. means to abort if encountering an error
&domains		domain definition: dimensions, nesting parameters
time_step	60	time step for integration in integer seconds (recommended 6*dx in km for a typical case)
time_step_fract_num	0	numerator for fractional time step
time_step_fract_den	1	denominator for fractional time step Example, if you want to use 60.3 sec as your time step, set time_step = 60, time_step_fract_num = 3, and time_step_fract_den = 10
time_step_dfi	60	time step for DFI, may be different from the regular time_step

max_dom	1	number of domains - set it to > 1 if it is a nested run
s_we (max_dom)	1	start index in x (west-east) direction (leave as is)
e_we (max_dom)	91	end index in x (west-east) direction (staggered dimension)
s_sn (max_dom)	1	start index in y (south-north) direction (leave as is)
e_sn (max_dom)	82	end index in y (south-north) direction (staggered dimension)
s_vert (max_dom)	1	start index in z (vertical) direction (leave as is)
e_vert (max_dom)	28	end index in z (vertical) direction (staggered dimension - this refers to full levels). Most variables are on unstaggered levels. Vertical dimensions need to be the same for all nests.
dx (max_dom)	10000	grid length in x-direction, unit in meters
dy (max_dom)	10000	grid length in y-direction, unit in meters
ztop (max_dom)	19000.	height in meters; used to define model top for idealized cases
grid_id (max_dom)	1	domain identifier
parent_id (max_dom)	0	id of the parent domain
i_parent_start (max_dom)	1	starting LLC I-indices from the parent domain
j_parent_start (max_dom)	1	starting LLC J-indices from the parent domain
parent_grid_ratio (max_dom)	1	parent-to-nest domain grid size ratio: for real-data cases the ratio has to be odd; for idealized cases, the ratio can be even if feedback is set to 0.
parent_time_step_ratio (max_dom)	1	parent-to-nest time step ratio; it can be different from the parent_grid_ratio
feedback	1	feedback from nest to its parent domain; 0 = no feedback
smooth_option	0	smoothing option for parent domain, used only with feedback option on. 0: no smoothing; 1: 1-2-1 smoothing; 2: smoothing-desmoothing
hypsometric_opt	2	If set to 2, it uses an alternative way (less biased when compared agaist input data) to compute height in program real and pressure in model (ARW only). 1: original way.
(options for program real)		
num_metgrid_levels	40	number of vertical levels in WPS output: type `ncdump -h` to find out

num_metgrid_soil_levels	4	number of soil levels or layers in WPS output
eta_levels	1.0, 0.99,…0.0	model *eta* levels from 1 to 0. If not given, *real* will provide a set of levels
force_sfc_in_vinterp	1	use surface data as lower boundary when interpolating through this many eta levels
interp_theta	.false.	If set to .false., it will vertically interpolate temp instead of potential temp (.true.), which may reduce bias when compared with input data
p_top_requested	5000	p_top to use in the model; must be available in WPS data
interp_type	2	vertical interpolation; 1: linear in pressure; 2: linear in log(pressure)
extrap_type	2	vertical extrapolation of non-temperature variables. 1: extrapolate using the two lowest levels; 2: use lowest level as constant below ground
t_extrap_type	2	vertical extrapolation for potential temperature. 1: isothermal; 2: -6.5 K/km lapse rate for temperature 3: constant theta
use_levels_below_ground	.true.	in vertical interpolation, whether to use levels below input surface level: true: use input isobaric levels below input surface false: extrapolate when WRF location is below input surface level
use_surface	.true.	whether to use input surface level data in vertical interpolation true: use input surface data false: do not use input surface data
lagrange_order	1	vertical interpolation order; 1: linear; 2: quadratic
lowest_lev_from_sfc	.false.	T = use surface values for the lowest *eta* (u,v,t,q); F = use traditional interpolation
sfcp_top_sfcp	.false.	optional method to compute model's surface pressure when incoming data only has surface pressure and terrain, but not SLP
use_tavg_for_tsk	.false.	whether to use diurnally-averaged surface temp as skin temp. The diurnally-averaged surface temp can be computed using WPS utility `avg_tsfc.exe`. May use this option when SKINTEMP is not present.

rh2qv_wrt_liquid	.true.	whether to compute Qv with respect to water (true) or ice (false)
smooth_cg_topo	.false.	smooth the outer rows and columns of the domain 1 topography w.r.t. the input data
vert_refine_fact	1	vertical refinement factor for ndown
(options for preset moving nest)		
num_moves	2,	total number of moves for all domains
move_id (max_moves)	2,2,	a list of nest domain id's, one per move
move_interval (max_moves)	60,120,	time in minutes since the start of this domain
move_cd_x (max_moves)	1,-1,	the number of parent domain grid cells to move in i-direction
move_cd_y (max_moves)	-1,1,	the number of parent domain grid cells to move in j-direction (positive in increasing i/j directions, and negative in decreasing i/j directions. Only 1, 0 and -1 is permitted.
(options for automatic moving nest)		
vortex_interval (max_dom)	15	how often the new vortex position is computed
max_vortex_speed (max_dom)	40	unit in m/sec; used to compute the search radius for the new vortex position
corral_dist (max_dom)	8	how many coarse grid cells the moving nest is allowed to get near the coarse grid boundary
track_level	50000.	Pressure level value (Pa) at which the tropical storm vortex is tracked
time_to_move (max_dom)	0.,	time, in minutes, to start moving nest
(options for adaptive time step)		
use_adaptive_time_step	.false.	whether to use adaptive time step
step_to_output_time	.true.	whether to modify the time steps so that the exact history time is reached
target_cfl(max_dom)	1.2. 1.2, 1.2,	if vertical CFL <= this value, then time step is increased
target_hcfl(max_dom)	0.84, 0.84, 0.84,	if horizontal CFL <= this value, then time step is increased
max_step_increase_pct(max_dom)	5, 51, 51,	percentage of previous time step to increase, if the max CFL is <= target_cfl
starting_time_step (max_dom)	-1, -1, -1,	flag -1 implies 6*dx is used to start the model. Any positive integer number specifies the time step the model will start with. Note that when use_adaptive_time_step is true, the value specified for time_step is ignored.
max_time_step(max_dom)	-1, -1, -1,	flag -1 implies the maximum time step is 3*starting_time_step. Any positive integer number specified the maximum time step

`min_time_step (max_dom)`	-1, -1, -1,	flag -1 implies the minimum time step is 0.5*starting_time_step. Any positive integer number specified the minumum time step
`adaptation_domain`	1	Which domain to drive adaptive time stepping. Default is domain 1.

(options to control parallel computing)

`tile_sz_x`	0	number of points in tile x direction
`tile_sz_y`	0	number of points in tile y direction; can be determined automatically
`numtiles`	1	number of tiles per patch (alternative to above two items)
`nproc_x`	-1	number of processors in x for decomposition
`nproc_y`	-1	number of processors in y for decomposition -1: code will do automatic decomposition >1: for both: will be used for decomposition
`&physics`		Physics options
`mp_physics (max_dom)`		*microphysics option*
	0	no microphysics
	1	Kessler scheme
	2	Lin et al. scheme
	3	WSM 3-class simple ice scheme
	4	WSM 5-class scheme
	5	Ferrier (new Eta) microphysics, operational High-Resolution Window
	6	WSM 6-class graupel scheme
	7	Goddard GCE scheme (also use `gsfcgce_hail` and `gsfcgce_2ice`)
	8	Thompson graupel scheme (2-moment scheme in V3.1)
	9	Milbrandt-Yau 2-moment scheme
	10	Morrison 2-moment scheme
	13	SBU-YLin, 5-class scheme
	14	WRF double moment, 5-class scheme
	16	WRF double moment, 6-class scheme
	17	NSSL 2-moment
	18	NSSL 2-moment, with CCN prediction
	95	Ferrier (old Eta), operational NAM (WRF NMM)
	98	Thompson scheme in V3.0

mp_zero_out		For non-zero mp_physics options, this keeps moisture variables above a threshold value >= 0. An alternative (and better) way to keep moisture variables positive is to use moist_adv_opt.
	0	no action taken, no adjustment to any moisture field
	1	except for Qv, all other moisture arrays are set to zero if they fall below a critical value
	2	Qv >= 0 and all other moisture arrays are set to zero if they fall below a critical value
mp_zero_out_thresh	1.e-8	critical value for moisture variable threshold, below which moisture arrays (except for Qv) are set to zero (unit: kg/kg)
mp_tend_lim	10..	limit on temp tendency from microphysics latent heating when radar data assimilation is used
gsfcgce_hail	0	0: running gsfcgce scheme with graupel 1: running gsfcgce scheme with hail
gsfcgce_2ice	0	0: running gsfcgce scheme with snow, ice and graupel / hail 1: running gsfcgce scheme with only ice and snow 2: running gsfcgce scheme with only ice and graupel (used only in very extreme situation)
no_mp_heating	0	switch to turn off latent heating from mp 0: normal 1: turn off latent heating from a microphysics scheme
ra_lw_physics (max_dom)		longwave radiation option
	0	no longwave radiation
	1	rrtm scheme
	3	CAM scheme
	4	rrtmg scheme
	5	Goddard scheme
	7	FLG (UCLA) scheme
	99	GFDL (Eta) longwave (semi-supported)
ra_sw_physics (max_dom)		shortwave radiation option
	0	no shortwave radiation
	1	Dudhia scheme
	2	(old) Goddard shortwave scheme
	3	CAM scheme
	4	rrtmg scheme
	5	Goddard scheme
	7	FLG (UCLA) scheme

	99	GFDL (Eta) longwave (semi-supported)
`radt (max_dom)`	30	minutes between radiation physics calls. Recommend 1 minute per km of dx (e.g. 10 for 10 km grid); use the same value for all nests
`co2tf`	1	CO2 transmission function flag for GFDL radiation only. Set it to 1 for ARW, which allows generation of CO2 function internally
`cam_abs_freq_s`	21600	CAM clear sky longwave absorption calculation frequency (recommended minimum value to speed scheme up)
`levsiz`	59	for CAM radiation input ozone levels
`paerlev`	29	for CAM radiation input aerosol levels
`cam_abs_dim1`	4	for CAM absorption save array
`cam_abs_dim2`	same as e_vert	for CAM 2nd absorption save array. The above 5 variables for CAM are automatically set in V3.2.
`sf_sfclay_physics (max_dom)`		surface-layer option
	0	no surface-layer
	1	MM5 Monin-Obukhov scheme
	2	Monin-Obukhov (Janjic Eta) scheme
	3	NCEP GFS scheme (NMM only)
	4	QNSE
	5	MYNN
	7	Pleim-Xiu (ARW only), only tested with Pleim-Xiu surface and ACM2 PBL
	10	TEMF
	11	Revised MM5 surface layer scheme
`iz0tlnd`	0	thermal roughness length for sfclay and myjsfc (0 - old, 1 - veg dependent Czil)
`sf_surface_physics (max_dom)`		land-surface option (set before running *real*; also set correct `num_soil_layers`)
	0	no surface temp prediction
	1	thermal diffusion scheme
	2	unified Noah land-surface model
	3	RUC land-surface model
	4	Noah-MP land-surface model (additional options under &noah_mp
	7	Pleim-Xiu scheme (ARW only)
	8	SSiB land-surface model (ARW only). Works with *ra_lw_physics* = 1, 3, 4 and *ra_sw_physics* = 1, 3, 4
`sf_urban_physics (max_dom)`		urban physics option (replacing `ucmcall` option in previous versions); works with Noah LSM

	0	no urban physics
	1	single-layer UCM (Kusaka)
	2	multi-layer, BEP (Martilli); works with BouLac and MYJ PBL only.
bl_pbl_physics (max_dom)		boundary-layer option
	0	no boundary-layer
	1	YSU scheme, use sf_sfclay_physics=1
	2	Mellor-Yamada-Janjic (Eta) TKE scheme, use sf_sfclay_physics=2
	3	NCEP GFS scheme (NMM only), use sf_sfclay_physics=3
	4	QNSE-EDMF, use sf_sfclay_physics=4
	5	MYNN 2.5 level TKE, use sf_sfclay_physics=1,2, and 5
	6	MYNN 3rd level TKE, use sf_sfclay_physics=5
	7	ACM2 (Pleim) scheme, use sf_sfclay_physics=1, 7
	8	Bougeault and Lacarrere (BouLac) TKE, use sf_sfclay_physics=1, 2
	9	Bretherton-Park/UW TKE scheme, use sf_sfclay_physics = 1, 2
	10	TEMF
	99	MRF scheme (to be removed)
mfshconv	1	turn on day-time EDMF for QNSE; 0=off
bldt (max_dom)	0	minutes between boundary-layer physics calls. 0 = call every time step
topo_wind (max_dom)	0	1 = turn on topographic surface wind correction (require extra input from geogrid, and works with YSU only)
grav_settling (max_dom)	0	Gravitational settling of fog/cloud droplet, MYNN PBL only
cu_physics (max_dom)		cumulus option
	0	no cumulus
	1	Kain-Fritsch (new Eta) scheme
	2	Betts-Miller-Janjic scheme
	3	Grell-Devenyi ensemble scheme
	4	Old Simplied Arakawa-Schubert (SAS)
	5	New Grell scheme (G3)
	6	Tiedtke scheme
	7	Zhang-McFarlane from CESM (works with MYJ and UW PBL)
	14	New GFS SAS from YSU
	84	New SAS (HWRF)
	99	previous Kain-Fritsch scheme

cudt	0	minutes between cumulus physics calls. 0 = call every time step
kfeta_trigger	1	KF trigger function option: = 1, default; = 2, moisture-adv; = 3, RH
ishallow	0	Shallow convection used with Grell 3D
shcu_physics(max_dom)	2	Bretherton-Park/UW
maxiens	1	Grell-Devenyi and G3 only
maxens	3	G-D only
maxens2	3	G-D only
maxens3	16	G-D only
ensdim	144	G-D only. These are recommended numbers. If you would like to use any other number, consult the code to understand what you are doing.
cugd_avedx	1	number of grid boxes over which subsidence is spread. default=1, for large grid sizes; use 3 for small grid sizes (<5km)
cu_diag (max_dom)	0	Additional time-averaged diagnostics from cu physics (G3 only)
convtrans_avglen_m	30	Averaging time for convective transport output variables (minutes, G3 only)
isfflx	1	heat and moisture fluxes from the surface 1 = with fluxes from the surface 0 = no flux from the surface (not for sf_surface_sfclay = 2). If diff_opt=2, km_opt=2 or 3 then 0 = constant fluxes defind by tke_drag_coefficient, tke_heat_flux; 1 = use model computed u*, and heat and moisture fluxes; 2 = use model computed u*, and specified heat flux by tke_heat_flux
ifsnow	0	snow-cover effects (only works for *sf_surface_physics* = 1) 1 = with snow-cover effect 0 = without snow-cover effect
icloud	1	cloud effect to the optical depth in radiation (only works for *ra_sw_physics* = 1, 4 and *ra_lw_physics* = 1, 4) 1 = with cloud effect 0 = without cloud effect
swrad_scat	1.	Scattering tuning parameter (default 1 is 1.e-5 m^2/kg) (only for *ra_sw_physics* = 1)
surface_input_source	1,2,3	where landuse and soil category data come from: 1 = WPS/geogrid, but with dominant categories recomputed in real

		2 = GRIB data from another model (only if arrays VEGCAT/SOILCAT exist) 3 = use dominant land and soil categories from WPS/geogrid
num_soil_layers		number of soil layers in land surface model (set in *real*)
	5	thermal diffusion scheme for temp only
	4	Noah land-surface model
	6	RUC land-surface model
	2	Pleim-Xu land-surface model
pxlsm_smois_init (max_dom)	1	PX LSM soil moisture initialization option 0: from analysis 1: from LANDUSE.TBL (SLMO)
num_land_cat	24	number of landuse categories in input data. 24 – USGS; 20 – MODIS; 28 – USGS+lake; 21 – MODIS+1
num_soil_cat	16	number of soil categories in input data
usemonalb	.false.	whether to use monthly albedo map instead of table values. Recommended for `sst_update = 1`
rdmaxalb	.true.	use snow albedo from geogrid; false means use snow albedo from table
rdlai2d	.false.	use LAI from input data; false means using values from table
seaice_threshold	271.	tsk < seaice_threshold, if water point and 5-layer slab scheme, set to land point and permanent ice; if water point and Noah scheme, set to land point, permanent ice, set temps from 3 m to surface, and set smois and sh2o
sst_update		option to use time-varying SST, seaice, vegetation fraction, and albedo during a model simulation (set before running *real*)
	0	no SST update
	1	*real.exe* will create `wrflowinp` file(s) at the same time interval as the available input data. Also set `auxinput4_inname = "wrflowinp_d<domain>"`, `auxinput4_interval` and `io_form_auxinput4` (required in V3.2) in namelist section *&time_control*
tmn_update	1	update deep layer soil temperature, useful for long simulations
lagday	150	days over which tmn is computed using skin temperature
sst_skin	1	calculate skin SST, useful for long simulations

bucket_mm	-1.	bucket reset values for water accumulation (unit in mm), useful for long simulations; -1 = inactive
bucket_j	-1.	bucket reset value for energy accumulations (unit in Joules) useful for long simulations; -1 = inactive
slope_rad (max_dom)	0	slope effects for ra_sw_physics (1=on, 0=off)
topo_shading (max_dom)	0	neighboring-point shadow effects for ra_sw_physics (1=on, 0=off)
shadlen	25000.	max shadow length in meters for topo_shading = 1
omlcall	0	simple ocean mixed layer model (1=on, 0=off)
oml_hml0	50.	>= 0: initial ocean mixed layer depth (m), constant everywhere < 0: use input
oml_gamma	0.14	lapse rate in deep water for oml (K m-1)
isftcflx	0	alternative Ck, Cd for tropical storm application. Ck: 0=off, 1=constant Z_{0q}, 2=Garratt. Cd: Donelan
fractional_seaice	0.	treat seaice as fractional field (1) or ice/no ice flag (0)
seaice_albedo_opt	0	Option to set albedo over sea ice. 0 = seaice albedo is a constant of 0.80; = 1 seaice albedo is a function of air temp, skin temp and snow.
prec_acc_dt	0.	number of minutes in precipitation bucket if set greater than 0.
(for wind turbine drag use)		
windturbines_spec	"none"	none, ideal, or a file name
td_turbgridid	-1	which grid id has turbines in it
td_hubheight	100.	hub height (m)
td_diameter	60.	turbine diameter (m)
td_stdthrcoef	.158	standing thrust coefficient
td_cutinspeed	4.	cut-in speed (m/s)
td_cutoutspeed	27.	cut-out speed (m/s)
td_power	2.	turbine power (MW)
td_turbpercell	1.	number of turbine per cell
td_ewfx	0	extend of wind farm in x-cells
td_ewfy	0	extend of wind farm in y-cells
td_pwfx	1	southwest corner of wind farm in x-cells
td_pwfy	1	southwest corner of wind farm in y-cells
(for stochastic kinetic-energy backscatter scheme [SKEB] (used to perturb a forecast)		
stoch_force_opt (max_dom)	0	whether to turn on SKEB: 0 – off; 1 - on

stoch_vertstruc_opt (max_dom)	0	0 - constant vertical structure of random pattern generator 1 - random phase vertical structure random pattern generator
tot_backscat_psi	1.0E-5	controls amplitude of rotational wind perturbations
tot_backscat_t	1.0E-6	controls amplitude of potential temperature perturbations
nens	1	controls random number stream
&noah_mp		Options for Noah-MP LSM
dveg	2	Dynamic vegetation option: 0 = off (LAI from table; FVEG=shdfac) 1 = on 2 = off (LAI from table; FVEG calculated 3 = off (LAI from table; FVEG = max veg fraction
opt_crs	1	stomatal resistance option: 1 = Ball-Berry; 2 = Jarvis
opt_sfc	1	surface layer drag coefficient calculation: 1 = Monin-Obukhov, 2 = original Noah, 3 = MYJ consistent, 4 = YSU consistent
opt_btr	1	soil moisture factor for stomatal resistance 1 = Noah, 2 = CLM, 3 = SSiB
opt_run	1	runoff and groundwater option: 1 = TOPMODEL with groundwater, 2 = TOPMODEL with equilibrium water table 3 = original surface and subsurface runoff (free drainage) 4 = BATS surface and subsurface runoff (free drainage)
opt_frz	1	supercooled liquid water option: 1 = no iteration, 2 = Koren's iteration
opt_inf	1	soil permeability option: 1 = linear effect, more permeable, 2 = non-linear effect, less permeable
opt_rad	1	radiative transfer option: 1 = modified two-stream, 2 = two-stream applied to grid cell, 3 = two-stream applied to vegetated fraction

opt_alb	2	ground surface albedo option: 1 = BATS, 2 = CLASS
opt_snf	1	precipitation partitioning between snow and rain: 1 = Jordan (1991) 2 = BATS; snow when SFCTMP < TFRZ+2.2 3 = snow when SFCTMP < TFRZ
opt_tbot	1	soil temp lower boundary condition: 1 = zero heat flux 2 = TBOT at 8 m from input file
opt_stc	1	snow/soil temperature time scheme: 1 = semi-implicit, 2 = full-implicit

&fdda *for grid, obs and spectral nudging*

(for grid nudging)

grid_fdda (max_dom)	1	grid analysis nudging on (=0 off)
	2	spectral analysis nudging option
gfdda_inname	"wrffdda_d<domain>"	defined name in real
gfdda_interval (max_dom)	360	Time interval (min) between analysis times
gfdda_end_h (max_dom)	6	Time (h) to stop nudging after start of forecast
io_form_gfdda	2	analysis format (2 = netcdf)
fgdt (max_dom)	0	calculation frequency (in minutes) for analysis nudging. 0 = every time step, and this is recommended
fgdtzero	0	not active
	1	nudging tendencies are set to zero in between fdda calls
if_no_pbl_nudging_uv (max_dom)	0	1= no nudging of u and v in the pbl; 0= nudging in the pbl
if_no_pbl_nudging_t (max_dom)	0	1= no nudging of temp in the pbl; 0= nudging in the pbl
if_no_pbl_nudging_q (max_dom)	0	1= no nudging of qvapor in the pbl; 0= nudging in the pbl
if_no_pbl_nudging_ph (max_dom)	0	1= no nudging of ph in the pbl; 0= nudging in the pbl; only for spectral nudging
if_zfac_uv (max_dom)	0	0= nudge u and v in all layers, 1= limit nudging to levels above k_zfac_uv
k_zfac_uv	10	10=model level below which nudging is switched off for u and v
if_zfac_t (max_dom)	0	0= nudge temp in all layers, 1= limit nudging to levels above k_zfac_t
k_zfac_t	10	10=model level below which nudging is switched off for temp

if_zfac_q (max_dom)	0	0= nudge qvapor in all layers, 1= limit nudging to levels above k_zfac_q
k_zfac_q	10	10=model level below which nudging is switched off for water qvapor
if_zfac_ph (max_dom)	0	0= nudge ph in all layers, 1= limit nudging to levels above k_zfac_ph (spectral nudging only)
k_zfac_q	10	10=model level below which nudging is switched off for water ph (spectral nudging only)
guv (max_dom)	0.0003	nudging coefficient for u and v (sec-1)
gt (max_dom)	0.0003	nudging coefficient for temp (sec-1)
gq (max_dom)	0.0003	nudging coefficient for qvapor (sec-1)
gph (max_dom)	0.0003	nudging coefficient for ph (sec-1), spectral nudging only
dk_zfac_uv (max_dom)	1	depth in k between k_zfac_X to dk_zfac_X where nudging increases linearly to full strength (spectral nudging only)
dk_zfac_t (max_dom)	1	
dk_zfac_ph (max_dom)	1	
xwavenum	3	top wave number to nudge in x direction, spectral nudging only
ywavenum	3	top wave number to nudge in y direction, spectral nudging only
if_ramping	0	0= nudging ends as a step function, 1= ramping nudging down at end of period
dtramp_min	60.	time (min) for ramping function, 60.0=ramping starts at last analysis time, -60.0=ramping ends at last analysis time
grid_sfdda (max_dom)	1	surface grid-nudging on (=0 off)
sgfdda_inname	“wrfsfdda_d<domain>”	defined name for surface nudging input file (from program obsgrid)
sgfdda_interval (max_dom)	360	time interval (min) between surface analysis times
sgfdda_end_h (max_dom)	6	time (in hours) to stop nudging after start of forecast
io_form_sgfdda	2	surface analysis format (2 = netcdf)
guv_sfc (max_dom)	0.0003	nudging coefficient for u and v (sec-1)
gt_sfc (max_dom)	0.0003	nudging coefficient for temp (sec-1)
gq_sfc (max_dom)	0.0003	nudging coefficient for qvapor (sec-1)
rinblw	250.	radius of influence used to determine the confidence (or weights) for the analysis, which is based on the distance between the grid point to the nearest obs. The analysis without nearby observation is used at a reduced weight

(for obs nudging)

obs_nudge_opt (max_dom)	1	obs-nudging fdda on (=0 off) for each domain; also need to set auxinput11_interval and auxinput11_end_h in time_control namelist
max_obs	150000	max number of observations used on a domain during any given time window
fdda_startj(max_dom)	0.	obs nudging start time in minutes
fdda_end (max_dom)	180.	obs nudging end time in minutes
obs_nudge_wind (max_dom)	1	whether to nudge wind: (=0 off)
obs_coef_wind (max_dom)	6.e-4	nudging coefficient for wind, unit: s-1
obs_nudge_temp (max_dom)	1	whether to nudge temperature: (=0 off)
obs_coef_temp (max_dom)	6.e-4	nudging coefficient for temp, unit: s-1
obs_nudge_mois (max_dom)	1	whether to nudge water vapor mixing ratio: (=0 off)
obs_coef_mois (max_dom)	6.e-4	nudging coefficient for water vapor mixing ratio, unit: s-1
obs_nudge_pstr (max_dom)	0	whether to nudge surface pressure (not used)
obs_coef_pstr (max_dom)	0.	nudging coefficient for surface pressure, unit: s-1 (not used)
obs_rinxy	200.	horizontal radius of influence in km
obs_rinsig	0.1	vertical radius of influence in *eta*
obs_twindo (max_dom)	0.666667	half-period time window over which an observation will be used for nudging; the unit is in hours
obs_npfi	10	freq in coarse grid timesteps for diag prints
obs_ionf (max_dom)	2	freq in coarse grid timesteps for obs input and err calc
obs_idynin	0	for dynamic initialization using a ramp-down function to gradually turn off the FDDA before the pure forecast (=1 on)
obs_dtramp	40.	time period in minutes over which the nudging is ramped down from one to zero.
obs_prt_max	10	maximum allowed obs entries in diagnostic printout
obs_prt_freq (max_dom)	10	frequency in obs index for diagnostic printout
obs_ipf_in4dob	.true.	print obs input diagnostics (=.false. off)
obs_ipf_errob	.true.	print obs error diagnostics (=.false. off)
obs_ipf_nudob	.true.	print obs nudge diagnostics (=.false. off)
obs_ipf_init	.true.	enable obs init warning messages
obs_no_pbl_nudge_uv (max_dom)	0	1= no wind-nudging within pbl
obs_no_pbl_nudge_t (max_dom)	0	1= no temperature-nudging within pbl

obs_no_pbl_nudge_q (max_dom)	0	1= no moisture-nudging within pbl
obs_nudgezfullr1_uv	50	Vert infl full weight height for LML obs, regime 1, winds
obs_nudgezrampr1_uv	50	Vert infl ramp-to-zero height for LML obs, regime 1, winds
obs_nudgezfullr2_uv	50	Vert infl full weight height for LML obs, regime 2, winds
obs_nudgezrampr2_uv	50	Vert infl ramp-to-zero height for LML obs, regime 2, winds
obs_nudgezfullr4_uv	-5000	Vert infl full weight height for LML obs, regime 4, winds
obs_nudgezrampr4_uv	50	Vert infl ramp-to-zero height for LML obs, regime 4, winds
obs_nudgezfullr1_t	50	Vert infl full weight height for LML obs, regime 1, temperature
obs_nudgezrampr1_t	50	Vert infl ramp-to-zero height for LML obs, regime 1, temperature
obs_nudgezfullr2_t	50	Vert infl full weight height for LML obs, regime 2, temperature
obs_nudgezrampr2_t	50	Vert infl ramp-to-zero height for LML obs, regime 2, temperature
obs_nudgezfullr4_t	-5000	Vert infl full weight height for LML obs, regime 4, temperature
obs_nudgezrampr4_t	50	Vert infl ramp-to-zero height for LML obs, regime 4, temperature
obs_nudgezfullr1_q	50	Vert infl full weight height for LML obs, regime 1, moisture
obs_nudgezrampr1_q	50	Vert infl ramp-to-zero height for LML obs, regime 1, moisture
obs_nudgezfullr2_q	50	Vert infl full weight height for LML obs, regime 2, moisture
obs_nudgezrampr2_q	50	Vert infl ramp-to-zero height for LML obs, regime 2, moisture
obs_nudgezfullr4_q	-50000	Vert infl full weight height for LML obs, regime 4, moisture
obs_nudgezrampr4_q	50	Vert infl ramp-to-zero height for LML obs, regime 4, moisture
obs_nudgezfullmin	50	Min depth through which vertical infl fcn remains 1.0
obs_nudgezrampmin	50	Min depth (m) through which vert infl fcn decreases from 1 to 0
obs_nudgezmax	3000	Max depth (m) in which vert infl function is nonzero
obs_sfcfact	1.0	Scale factor applied to time window for surface obs
obs_sfcfacr	1.0	Scale factor applied to horiz radius of influence for surface obs

obs_dpsmx	7.5	Max pressure change (cb) allowed within horiz radius of influence
obs_sfc_scheme_horiz	0	horizontal spreading scheme for surface obs: 0 – WRF scheme; 1 – MM5 scheme
obs_sfc_scheme_vert	0	vertical spreading scheme for surface obs: 0 – regime vif; 1 – original scheme
obs_max_sndng_gap	20	max allowed pressure gap in soundings for interp (in centibars)
&dynamics		*Diffusion, damping options, advection options*
rk_ord		time-integration scheme option:
	2	Runge-Kutta 2nd order
	3	Runge-Kutta 3rd order (recommended)
diff_opt		turbulence and mixing option:
	0	= no turbulence or explicit spatial numerical filters (km_opt IS IGNORED).
	1	evaluates 2nd order diffusion term on coordinate surfaces. uses kvdif for vertical diff unless PBL option is used. may be used with km_opt = 1 and 4. (= 1, recommended for real-data case)
	2	evaluates mixing terms in physical space (stress form) (x,y,z). turbulence parameterization is chosen by specifying km_opt.
km_opt		eddy coefficient option
	1	constant (use khdif and kvdif)
	2	1.5 order TKE closure (3D)
	3	Smagorinsky first order closure (3D) Note: option 2 and 3 are not recommended for DX > 2 km
	4	horizontal Smagorinsky first order closure (recommended for real-data case)
diff_6th_opt (max_dom)	0	6th-order numerical diffusion 0 = no 6th-order diffusion (default) 1 = 6th-order numerical diffusion 2 = 6th-order numerical diffusion but prohibit up-gradient diffusion
diff_6th_factor (max_dom)	0.12	6th-order numerical diffusion non-dimensional rate (max value 1.0 corresponds to complete removal of 2dx wave in one timestep)
damp_opt		upper level damping flag
	0	without damping

	1	with diffusive damping; maybe used for real-data cases (`dampcoef` nondimensional ~ 0.01 - 0.1)
	2	with Rayleigh damping (`dampcoef` inverse time scale [1/s], e.g. 0.003)
	3	with w-Rayleigh damping (`dampcoef` inverse time scale [1/s] e.g. 0.2; for real-data cases)
`zdamp (max_dom)`	5000	damping depth (m) from model top
`dampcoef (max_dom)`	0.	damping coefficient (see `damp_opt`)
`w_damping`		vertical velocity damping flag (for operational use)
	0	without damping
	1	with damping
`base_pres`	100000.	Base state surface pressure (Pa), real only. Do not change.
`base_temp`	290.	Base state sea level temperature (K), real only.
`base_lapse`	50.	real-data ONLY, lapse rate (K), DO NOT CHANGE.
`iso_temp`	0.	isothermal temperature in stratosphere, `real` only, enable the model to be extended to 5 mb
`use_baseparm_fr_nml`	.false.	for backward compatibility: to use with old wrfinput file
`khdif (max_dom)`	0	horizontal diffusion constant (m^2/s)
`kvdif (max_dom)`	0	vertical diffusion constant (m^2/s)
`smdiv (max_dom)`	0.1	divergence damping (0.1 is typical)
`emdiv (max_dom)`	0.01	external-mode filter coef for mass coordinate model (0.01 is typical for real-data cases)
`epssm (max_dom)`	.1	time off-centering for vertical sound waves
`non_hydrostatic (max_dom)`	.true.	whether running the model in hydrostatic or non-hydro mode
`pert_coriolis (max_dom)`	.false.	Coriolis only acts on wind perturbation (idealized)
`top_lid (max_dom)`	.false.	zero vertical motion at top of domain (idealized)
`mix_full_fields`	.false.	used with diff_opt = 2; value of ".true." is recommended, except for highly idealized numerical tests; `damp_opt` must not be 1 if ".true." is chosen. .false. means subtract 1-d base-state profile before mixing (idealized)
`mix_isotropic(max_dom)`	0	0=anistropic vertical/horizontal diffusion coeffs, 1=isotropic, for km_opt = 2, 3

mix_upper_bound(max_dom)	0.1	non-dimensional upper limit for diffusion coeffs, for km_opt = 2, 3
h_mom_adv_order (max_dom)	5	horizontal momentum advection order (5=5th, etc.)
v_mom_adv_order (max_dom)	3	vertical momentum advection order
h_sca_adv_order (max_dom)	5	horizontal scalar advection order
v_sca_adv_order (max_dom)	3	vertical scalar advection order
time_step_sound (max_dom)	4	number of sound steps per time-step (if using a time_step much larger than 6*dx (in km), increase number of sound steps). = 0: the value computed automatically
moist_adv_opt (max_dom)	1	advection options for moisture: 0= none positive-definite 1= positive-definite 2= monotonic 3= 5th order WENO 4= 5th WENO with positive definite
scalar_adv_opt (max_dom)	1	advection options for scalars: 0= none positive-definite 1= positive-definite 2= monotonic 3= 5th WENO 4= 5th WENO with positive definite
tke_adv_opt (max_dom)	1	advection options for tke: 0= none positive-definite 1= positive-definite 2= monotonic 3= 5th WENO 4= 5th WENO with positive definite
chem_adv_opt (max_dom)	1	advection options for chem vars: 0= none positive-definite 1= positive-definite 2= monotonic 3= 5th WENO 4= 5th WENO with positive definite
tracer_adv_opt (max_dom)	1	advection options for chem vars: 0= none positive-definite 1= positive-definite 2= monotonic 3= 5th WENO 4= 5th WENO with positive definite
momentum_adv_opt	1	advection options for momentum: 1= original 3= 5th order WENO
tke_drag_coefficient (max_dom)	0	surface drag coefficient (Cd, dimensionless) for diff_opt=2 only

tke_heat_flux (max_dom)	0	surface thermal flux (H/rho*cp), K m/s) for diff_opt = 2 only
fft_filter_lat	45.	the latitude above which the polar filter is turned on for global model
gwd_opt	0	gravity wave drag option (1= on), use when grid size > 10 km
do_avgflx_em (max_dom)	0	whether to output time-averaged mass-coupled advective velocities
do_avgflx_cugd	0	whether to output time-averaged convective mass-fluxes from Grell-Devenyi ensemble scheme
sfs_opt (max_dom)	0	nonlinear backscatter and anisotropy (NBA); default off
	1	using diagnostic stress terms (km_opt=2,3 for scalars)
	2	using tke-based stress terms (km_opt=2 needed)
m_opt (max_dom)	0	=1: adds output of Mij stress terms when NBA is not used
tracer_opt (max_dom)	0	=2: activate 8 pre-defined tracers in Registry
rad_nudge	0	option to nudge toward initial sounding in idealized TC case

&bdy_control *boundary condition control*

spec_bdy_width	5	total number of rows for specified boundary value nudging
spec_zone	1	number of points in specified zone (spec b.c. option)
relax_zone	4	number of points in relaxation zone (spec b.c. option)
specified (max_dom)	.false.	specified boundary conditions (only can be used for to domain 1)
spec_exp	0.	exponential multiplier for relaxation zone ramp for specified=.t. (0.= linear ramp default; 0.33=~3*dx exp decay factor)
		The above 5 namelists are used for real-data runs only
periodic_x (max_dom)	.false.	periodic boundary conditions in x direction
symmetric_xs (max_dom)	.false.	symmetric boundary conditions at x start (west)
symmetric_xe (max_dom)	.false.	symmetric boundary conditions at x end (east)
open_xs (max_dom)	.false.	open boundary conditions at x start (west)
open_xe (max_dom)	.false.	open boundary conditions at x end (east)
periodic_y (max_dom)	.false.	periodic boundary conditions in y direction

symmetric_ys (max_dom)	.false.	symmetric boundary conditions at y start (south)
symmetric_ye (max_dom)	.false.	symmetric boundary conditions at y end (north)
open_ys (max_dom)	.false.	open boundary conditions at y start (south)
open_ye (max_dom)	.false.	open boundary conditions at y end (north)
nested (max_dom)	.false.,.true.,.true.,	nested boundary conditions (must be set to .true. for nests)
polar	.false.	polar boundary condition (v=0 at polarward-most v-point) for global application
constant_bc	.false.	constant boundary condition used with DFI.
&namelist_quilt		*Option for asynchronized I/O for MPI applications*
nio_tasks_per_group	0	default value is 0: no quilting; > 0: the number of processors used for IO quilting per IO group
nio_groups	1	default 1. Maybe set to higher value for nesting IO, or history and restart IO
&grib2		
background_proc_id	255	Background generating process identifier, typically defined by the originating center to identify the background data that was used in creating the data. This is octet 13 of Section 4 in the grib2 message
forecast_proc_id	255	Analysis or generating forecast process identifier, typically defined by the originating center to identify the forecast process that was used to generate the data. This is octet 14 of Section 4 in the grib2 message
production_status	255	Production status of processed data in the grib2 message. See Code Table 1.3 of the grib2 manual. This is octet 20 of Section 1 in the grib2 record
compression	40	The compression method to encode the output grib2 message. Only 40 for jpeg2000 or 41 for PNG are supported
dfi_radar	0	DFI radar data assimilation switch
&dfi_control	digital filter option control (does not yet support nesting)	
dfi_opt	3	which DFI option to use 0: no digital filter initialization 1: digital filter launch (DFL)

		2: diabatic DFI (DDFI) 3: twice DFI (TDFI) (recommended)
dfi_nfilter	7	digital filter type: 0 – uniform; 1- Lanczos; 2 – Hamming; 3 – Blackman; 4 – Kaiser; 5 – Potter; 6 – Dolph window; 7 – Dolph (recommended); 8 – recursive high-order
dfi_write_filtered_input	.true.	whether to write wrfinput file with filtered model state before beginning forecast
dfi_write_dfi_history	.false.	whether to write wrfout files during filtering integration
dfi_cutoff_seconds	3600	cutoff period, in seconds, for the filter. Should not be longer than the filter window
dfi_time_dim	1000	maximum number of time steps for filtering period, this value can be larger than necessary
dfi_bckstop_year	2001	four-digit year of stop time for backward DFI integration. For a model that starts from 2001061112, this specifies 1 hour backward integration
dfi_bckstop_month	06	two-digit month of stop time for backward DFI integration
dfi_bckstop_day	11	two-digit day of stop time for backward DFI integration
dfi_bckstop_hour	11	two-digit hour of stop time for backward DFI integration
dfi_bckstop_minute	00	two-digit minute of stop time for backward DFI integration
dfi_bckstop_second	00	two-digit second of stop time for backward DFI integration
dfi_fwdstop_year	2001	four-digit year of stop time for forward DFI integration. For a model that starts at 2001061112, this specifies 30 minutes of forward integration
dfi_fwdstop_month	06	two-digit month of stop time for forward DFI integration
dfi_fwdstop_day	11	two-digit day of stop time for forward DFI integration
dfi_fwdstop_hour	12	two-digit hour of stop time for forward DFI integration
dfi_fwdstop_minute	30	two-digit minute of stop time for forward DFI integration
dfi_fwdstop_second	00	two-digit second of stop time for forward DFI integration
dfi_radar	0	DFI radar DA switch
&scm		*for single column model option only*

scm_force	1	switch for single column forcing (=0 off)
scm_force_dx	4000.	DX for SCM forcing (in meters)
num_force_layers	8	number of SCM input forcing layers
scm_lu_index	2	SCM landuse category (2 is dryland, cropland and pasture)
scm_isltyp	4	SCM soil category (4 is silt loam)
scm_vegfra	0.5	SCM vegetation fraction
scm_canwat	0.0	SCM canopy water
scm_lat	37.	SCM latitude
scm_lon	-96.	SCM longitude
scm_th_adv	.true.	turn on theta advection in SCM
scm_wind_adv	.true.	turn on wind advection in SCM
scm_qv_adv	.true.	turn on moisture advection in SCM
scm_vert_adv	.true.	turn on vertical advection in SCM
scm_ql_adv	.false.	turn on ql advection in SCM
num_force_soil_layers	5	number of SCM soil forcing layer
scm_soilt_force	.false.	turn on soil temp forcing in SCM
scm_soilq_force	.false.	turn on soil moisture forcing in SCM
scm_force_th_largescale	.false.	turn on largescale theta forcing in SCM
scm_force_qv_largescale	.false.	turn on largescale Qv forcing in SCM
scm_force_ql_largescale	.false.	turn on largescale ql forcing in SCM
scm_force_wind_largescale	.false.	turn on largescale wind forcing in SCM

&tc	*controls for tc_em.exe only*	
insert_bogus_storm	.false.	T/F for inserting a bogus tropical storm
remove_storm	.false.	T/F for only removing the original TC
num_storm	1	number of bogus TC
latc_loc	-999.	center latitude of the bogus TC
lonc_loc	-999.	center longitude of the bogus TC
vmax_meters_per second (max_dom)	-999.	vmax of bogus storm in meters per second
rmax	-999.	maximum radius outward from storm center
vmax_ratio (max_dom)	-999.	ratio for representative maximum winds, 0.75 for 45 km grid, and 0.9 for 15 km grid
rankine_lid	-999.	top pressure limit for the tc bogus scheme

WRF Output Fields

List of Fields

The following is an edited output list from the netCDF command *'ncdump -h'*. Note that valid output fields will depend on the model options used. If the fields have zero values, then the fields are not computed by the model options selected.

```
ncdump -h wrfout_d<domain>_<date>
```

netcdf wrfout_d01_2000-01-24_12:00:00

dimensions:

Time = UNLIMITED ; // (1 currently)
DateStrLen = 19 ;
west_east = 73 ;
south_north = 60 ;
bottom_top = 27 ;
bottom_top_stag = 28 ;
soil_layers_stag = 4 ;
west_east_stag = 74 ;
south_north_stag = 61 ;

variables:

char Times(Time, DateStrLen) ;
float LU_INDEX(Time, south_north, west_east) ;
LU_INDEX:description = "LAND USE CATEGORY" ;
LU_INDEX:units = "" ;
float ZNU(Time, bottom_top) ;
ZNU:description = "eta values on half (mass) levels" ;
ZNU:units = "" ;
float ZNW(Time, bottom_top_stag) ;
ZNW:description = "eta values on full (w) levels" ;
ZNW:units = "" ;
float ZS(Time, soil_layers_stag) ;
ZS:description = "DEPTHS OF CENTERS OF SOIL LAYERS" ;
ZS:units = "m" ;
float DZS(Time, soil_layers_stag) ;
DZS:description = "THICKNESSES OF SOIL LAYERS" ;
DZS:units = "m" ;
float U(Time, bottom_top, south_north, west_east_stag) ;
U:description = "x-wind component" ;
U:units = "m s-1" ;
float V(Time, bottom_top, south_north_stag, west_east) ;
V:description = "y-wind component" ;
V:units = "m s-1" ;
float W(Time, bottom_top_stag, south_north, west_east) ;
W:description = "z-wind component" ;
W:units = "m s-1" ;
float PH(Time, bottom_top_stag, south_north, west_east) ;
PH:description = "perturbation geopotential" ;
PH:units = "m2 s-2" ;
float PHB(Time, bottom_top_stag, south_north, west_east) ;
PHB:description = "base-state geopotential" ;
PHB:units = "m2 s-2" ;
float T(Time, bottom_top, south_north, west_east) ;
T:description = "perturbation potential temperature (theta-t0)" ;
T:units = "K" ;
float MU(Time, south_north, west_east) ;
MU:description = "perturbation dry air mass in column" ;
MU:units = "Pa" ;
float MUB(Time, south_north, west_east) ;
MUB:description = "base state dry air mass in column" ;
MUB:units = "Pa" ;
float NEST_POS(Time, south_north, west_east) ;
NEST_POS:description = "-" ;
NEST_POS:units = "-" ;
float P(Time, bottom_top, south_north, west_east) ;
P:description = "perturbation pressure" ;
P:units = "Pa" ;
float PB(Time, bottom_top, south_north, west_east) ;
PB:description = "BASE STATE PRESSURE" ;
PB:units = "Pa" ;

```
float FNM(Time, bottom_top) ;
      FNM:description = "upper weight for vertical stretching" ;
      FNM:units = "" ;
float FNP(Time, bottom_top) ;
      FNP:description = "lower weight for vertical stretching" ;
      FNP:units = "" ;
float RDNW(Time, bottom_top) ;
      RDNW:description = "inverse d(eta) values between full (w) levels" ;
      RDNW:units = "" ;
float RDN(Time, bottom_top) ;
      RDN:description = "inverse d(eta) values between half (mass) levels" ;
      RDN:units = "" ;
float DNW(Time, bottom_top) ;
      DNW:description = "d(eta) values between full (w) levels" ;
      DNW:units = "" ;
float DN(Time, bottom_top) ;
      DN:description = "d(eta) values between half (mass) levels" ;
      DN:units = "" ;
float CFN(Time) ;
      CFN:description = "extrapolation constant" ;
      CFN:units = "" ;
float CFN1(Time) ;
      CFN1:description = "extrapolation constant" ;
      CFN1:units = "" ;
 float P_HYD(Time, bottom_top, south_north, west_east) ;
      P_HYD:description = "hydrostatic pressure" ;
      P_HYD:units = "Pa" ;
float Q2(Time, south_north, west_east) ;
      Q2:description = "QV at 2 M" ;
      Q2:units = "kg kg-1" ;
float T2(Time, south_north, west_east) ;
      T2:description = "TEMP at 2 M" ;
      T2:units = "K" ;
float TH2(Time, south_north, west_east) ;
      TH2:description = "POT TEMP at 2 M" ;
      TH2:units = "K" ;
float PSFC(Time, south_north, west_east) ;
      PSFC:description = "SFC PRESSURE" ;
      PSFC:units = "Pa" ;
float U10(Time, south_north, west_east) ;
      U10:description = "U at 10 M" ;
      U10:units = "m s-1" ;
float V10(Time, south_north, west_east) ;
      V10:description = "V at 10 M" ;
      V10:units = "m s-1" ;
float RDX(Time) ;
      RDX:description = "INVERSE X GRID LENGTH" ;
      RDX:units = "" ;
float RDY(Time) ;
      RDY:description = "INVERSE Y GRID LENGTH" ;
      RDY:units = "" ;
float RESM(Time) ;
      RESM:description = "TIME WEIGHT CONSTANT FOR SMALL STEPS" ;
      RESM:units = "" ;
float ZETATOP(Time) ;
      ZETATOP:description = "ZETA AT MODEL TOP" ;
      ZETATOP:units = "" ;
float CF1(Time) ;
      CF1:description = "2nd order extrapolation constant" ;
      CF1:units = "" ;
float CF2(Time) ;
      CF2:description = "2nd order extrapolation constant" ;
      CF2:units = "" ;
float CF3(Time) ;
      CF3:description = "2nd order extrapolation constant" ;
      CF3:units = "" ;
```

```
int ITIMESTEP(Time) ;
        ITIMESTEP:description = "" ;
        ITIMESTEP:units = "" ;
float XTIME(Time) ;
        XTIME:description = "minutes since simulation start" ;
        XTIME:units = "" ;
float QVAPOR(Time, bottom_top, south_north, west_east) ;
        QVAPOR:description = "Water vapor mixing ratio" ;
        QVAPOR:units = "kg kg-1" ;
float QCLOUD(Time, bottom_top, south_north, west_east) ;
        QCLOUD:description = "Cloud water mixing ratio" ;
        QCLOUD:units = "kg kg-1" ;
float QRAIN(Time, bottom_top, south_north, west_east) ;
        QRAIN:description = "Rain water mixing ratio" ;
        QRAIN:units = "kg kg-1" ;
float LANDMASK(Time, south_north, west_east) ;
        LANDMASK:description = "LAND MASK (1 FOR LAND, 0 FOR WATER)" ;
        LANDMASK:units = "" ;
float TSLB(Time, soil_layers_stag, south_north, west_east) ;
        TSLB:description = "SOIL TEMPERATURE" ;
        TSLB:units = "K" ;
float SMOIS(Time, soil_layers_stag, south_north, west_east) ;
        SMOIS:description = "SOIL MOISTURE" ;
        SMOIS:units = "m3 m-3" ;
float SH2O(Time, soil_layers_stag, south_north, west_east) ;
        SH2O:description = "SOIL LIQUID WATER" ;
        SH2O:units = "m3 m-3" ;
float SEAICE(Time, south_north, west_east) ;
       SEAICE:description = "SEA ICE FLAG" ;
        SEAICE:units = "" ;
float XICEM(Time, south_north, west_east) ;
        XICEM:description = "SEA ICE FLAG (PREVIOUS STEP)" ;
        XICEM:units = "" ;
float SFROFF(Time, south_north, west_east) ;
        SFROFF:description = "SURFACE RUNOFF" ;
        SFROFF:units = "mm" ;
float UDROFF(Time, south_north, west_east) ;
        UDROFF:description = "UNDERGROUND RUNOFF" ;
        UDROFF:units = "mm" ;
int IVGTYP(Time, south_north, west_east) ;
        IVGTYP:description = "DOMINANT VEGETATION CATEGORY" ;
        IVGTYP:units = "" ;
int ISLTYP(Time, south_north, west_east) ;
        ISLTYP:description = "DOMINANT SOIL CATEGORY" ;
        ISLTYP:units = "" ;
float VEGFRA(Time, south_north, west_east) ;
        VEGFRA:description = "VEGETATION FRACTION" ;
        VEGFRA:units = "" ;
float GRDFLX(Time, south_north, west_east) ;
        GRDFLX:description = "GROUND HEAT FLUX" ;
        GRDFLX:units = "W m-2" ;
float SNOW(Time, south_north, west_east) ;
        SNOW:description = "SNOW WATER EQUIVALENT" ;
        SNOW:units = "kg m-2" ;
float SNOWH(Time, south_north, west_east) ;
        SNOWH:description = "PHYSICAL SNOW DEPTH" ;
        SNOWH:units = "m" ;
float RHOSN(Time, south_north, west_east) ;
        RHOSN:description = " SNOW DENSITY" ;
        RHOSN:units = "kg m-3" ;
float CANWAT(Time, south_north, west_east) ;
        CANWAT:description = "CANOPY WATER" ;
        CANWAT:units = "kg m-2" ;
float SST(Time, south_north, west_east) ;
        SST:description = "SEA SURFACE TEMPERATURE" ;
        SST:units = "K" ;
```

```
float SSTSK(Time, south_north, west_east) ;
      SSTSK:description = "SKIN SEA SURFACE TEMPERATURE" ;
      SSTSK:units = "K" ;
float MAPFAC_M(Time, south_north, west_east) ;
       MAPFAC_M:description = "Map scale factor on mass grid" ;
       MAPFAC_M:units = "" ;
float MAPFAC_U(Time, south_north, west_east_stag) ;
       MAPFAC_U:description = "Map scale factor on u-grid" ;
       MAPFAC_U:units = "" ;
float MAPFAC_V(Time, south_north_stag, west_east) ;
       MAPFAC_V:description = "Map scale factor on v-grid" ;
       MAPFAC_V:units = "" ;
float MAPFAC_MX(Time, south_north, west_east) ;
       MAPFAC_MX:description = "Map scale factor on mass grid, x direction" ;
       MAPFAC_MX:units = "" ;
float MAPFAC_MY(Time, south_north, west_east) ;
       MAPFAC_MY:description = "Map scale factor on mass grid, y direction" ;
       MAPFAC_MY:units = "" ;
float MAPFAC_UX(Time, south_north, west_east_stag) ;
       MAPFAC_UX:description = "Map scale factor on u-grid, x direction" ;
       MAPFAC_UX:units = "" ;
float MAPFAC_UY(Time, south_north, west_east_stag) ;
       MAPFAC_UY:description = "Map scale factor on u-grid, y direction" ;
       MAPFAC_UY:units = "" ;
float MAPFAC_VX(Time, south_north_stag, west_east) ;
       MAPFAC_VX:description = "Map scale factor on v-grid, x direction" ;
       MAPFAC_VX:units = "" ;
float MF_VX_INV(Time, south_north_stag, west_east) ;
       MF_VX_INV:description = "Inverse map scale factor on v-grid, x direction"
       MF_VX_INV:units = "" ;
float MAPFAC_VY(Time, south_north_stag, west_east) ;
       MAPFAC_VY:description = "Map scale factor on v-grid, y direction" ;
       MAPFAC_VY:units = "" ;
float F(Time, south_north, west_east) ;
       F:description = "Coriolis sine latitude term" ;
       F:units = "s-1" ;
float E(Time, south_north, west_east) ;
       E:description = "Coriolis cosine latitude term" ;
       E:units = "s-1" ;
float SINALPHA(Time, south_north, west_east) ;
       SINALPHA:description = "Local sine of map rotation" ;
       SINALPHA:units = "" ;
float COSALPHA(Time, south_north, west_east) ;
       COSALPHA:description = "Local cosine of map rotation" ;
       COSALPHA:units = "" ;
float HGT(Time, south_north, west_east) ;
       HGT:description = "Terrain Height" ;
       HGT:units = "m" ;
float HGT_SHAD(Time, south_north, west_east) ;
       HGT_SHAD:description = "Height of orographic shadow" ;
       HGT_SHAD:units = "m" ;
float TSK(Time, south_north, west_east) ;
       TSK:description = "SURFACE SKIN TEMPERATURE" ;
       TSK:units = "K" ;
float P_TOP(Time) ;
       P_TOP:description = "PRESSURE TOP OF THE MODEL" ;
       P_TOP:units = "Pa" ;
 float T00(Time) ;
      T00:description = "BASE STATE TEMPERATURE" ;
      T00:units = "K" ;
 float P00(Time) ;
      P00:description = "BASE STATE PRESURE" ;
      P00:units = "Pa" ;
 float TLP(Time) ;
      TLP:description = "BASE STATE LAPSE RATE" ;
      TLP:units = "" ;
```

```
 float TISO(Time) ;
       TISO:description = "TEMP AT WHICH THE BASE T TURNS CONST" ;
       TISO:units = "K" ;
float MAX_MSTFX(Time) ;
       MAX_MSTFX:description = "Max map factor in domain" ;
       MAX_MSTFX:units = "" ;
float MAX_MSTFY(Time) ;
       MAX_MSTFY:description = "Max map factor in domain" ;
       MAX_MSTFY:units = "" ;
float RAINC(Time, south_north, west_east) ;
       RAINC:description = "ACCUMULATED TOTAL CUMULUS PRECIPITATION" ;
       RAINC:units = "mm" ;
float RAINSH(Time, south_north, west_east) ;
      RAINSH:description = "ACCUMULATED SHALLOW CUMULUS PRECIPITATION" ;
      RAINSH:units = "mm" ;
float RAINNC(Time, south_north, west_east) ;
       RAINNC:description = "ACCUMULATED TOTAL GRID SCALE PRECIPITATION" ;
       RAINNC:units = "mm" ;
float PRATEC(Time, south_north, west_east) ;
       PRATEC:description = "PRECIP RATE FROM CUMULUS SCHEME" ;
       PRATEC:units = "mm s-1" ;
float RAINCV(Time, south_north, west_east) ;
       RAINCV:description = "TIME-STEP CUMULUS PRECIPITATION" ;
       RAINCV:units = "mm" ;
float SNOWNC(Time, south_north, west_east) ;
       SNOWNC:description = "ACCUMULATED TOTAL GRID SCALE SNOW AND ICE" ;
       SNOWNC:units = "mm" ;
float GRAUPELNC(Time, south_north, west_east) ;
       GRAUPELNC:description = "ACCUMULATED TOTAL GRID SCALE GRAUPEL" ;
       GRAUPELNC:units = "mm" ;
 float SWDOWN(Time, south_north, west_east) ;
       SWDOWN:description = "DOWNWARD SHORT WAVE FLUX AT GROUND SURFACE" ;
       SWDOWN:units = "W m-2" ;
float GLW(Time, south_north, west_east) ;
       GLW:description = "DOWNWARD LONG WAVE FLUX AT GROUND SURFACE" ;
       GLW:units = "W m-2" ;
float SWNORM(Time, south_north, west_east) ;
      SWNORM:description = "NORMAL SHORT WAVE FLUX AT GROUND SURFACE" ;
      SWNORM:units = "W m-2" ;
float OLR(Time, south_north, west_east) ;
       OLR:description = "TOA OUTGOING LONG WAVE" ;
       OLR:units = "W m-2" ;
float XLAT(Time, south_north, west_east) ;
       XLAT:description = "LATITUDE, SOUTH IS NEGATIVE" ;
       XLAT:units = "degree_north" ;
float XLONG(Time, south_north, west_east) ;
       XLONG:description = "LONGITUDE, WEST IS NEGATIVE" ;
       XLONG:units = "degree_east" ;
float XLAT_U(Time, south_north, west_east_stag) ;
       XLAT_U:description = "LATITUDE, SOUTH IS NEGATIVE" ;
       XLAT_U:units = "degree_north" ;
float XLONG_U(Time, south_north, west_east_stag) ;
       XLONG_U:description = "LONGITUDE, WEST IS NEGATIVE" ;
       XLONG_U:units = "degree_east" ;
float XLAT_V(Time, south_north_stag, west_east) ;
       XLAT_V:description = "LATITUDE, SOUTH IS NEGATIVE" ;
       XLAT_V:units = "degree_north" ;
float XLONG_V(Time, south_north_stag, west_east) ;
       XLONG_V:description = "LONGITUDE, WEST IS NEGATIVE" ;
       XLONG_V:units = "degree_east" ;
float ALBEDO(Time, south_north, west_east) ;
       ALBEDO:description = "ALBEDO" ;
       ALBEDO:units = "-" ;
float ALBBCK(Time, south_north, west_east) ;
       ALBBCK:description = "BACKGROUND ALBEDO" ;
       ALBBCK:units = "" ;
```

```
float EMISS(Time, south_north, west_east) ;
       EMISS:description = "SURFACE EMISSIVITY" ;
       EMISS:units = "" ;
 float NOAHRES(Time, south_north, west_east) ;
      NOAHRES:description = "RESIDUAL OF THE NOAH SURFACE ENERGY BUDGET" ;
      NOAHRES:units = "W m{-2}" ;
float TMN(Time, south_north, west_east) ;
       TMN:description = "SOIL TEMPERATURE AT LOWER BOUNDARY" ;
       TMN:units = "K" ;
float XLAND(Time, south_north, west_east) ;
       XLAND:description = "LAND MASK (1 FOR LAND, 2 FOR WATER)" ;
       XLAND:units = "" ;
float ZNT(Time, south_north, west_east) ;
      ZNT:description = "TIME-VARYING ROUGHNESS LENGTH" ;
      ZNT:units = "m" ;
float UST(Time, south_north, west_east) ;
       UST:description = "U* IN SIMILARITY THEORY" ;
       UST:units = "m s-1" ;
float PBLH(Time, south_north, west_east) ;
       PBLH:description = "PBL HEIGHT" ;
       PBLH:units = "m" ;
float HFX(Time, south_north, west_east) ;
       HFX:description = "UPWARD HEAT FLUX AT THE SURFACE" ;
       HFX:units = "W m-2" ;
float QFX(Time, south_north, west_east) ;
       QFX:description = "UPWARD MOISTURE FLUX AT THE SURFACE" ;
       QFX:units = "kg m-2 s-1" ;
float LH(Time, south_north, west_east) ;
       LH:description = "LATENT HEAT FLUX AT THE SURFACE" ;
       LH:units = "W m-2" ;
float SNOWC(Time, south_north, west_east) ;
       SNOWC:description = "FLAG INDICATING SNOW COVERAGE (1 FOR SNOW COVER)" ;
       SNOWC:units = "" ;
int SAVE_TOPO_FROM_REAL(Time) ;
      SAVE_TOPO_FROM_REAL:description = "1=original topo from real/0=topo
             modified by WRF" ;
     SAVE_TOPO_FROM_REAL:units = "flag" ;
```

List of Global Attributes

```
// global attributes:

            :TITLE = " OUTPUT FROM WRF V3.3 MODEL" ;
      :START_DATE = "2000-01-24_12:00:00" ;
      :SIMULATION_START_DATE = "2000-01-24_12:00:00" ;
      :WEST-EAST_GRID_DIMENSION = 74 ;
      :SOUTH-NORTH_GRID_DIMENSION = 61 ;
      :BOTTOM-TOP_GRID_DIMENSION = 28 ;
     :DX = 30000.f ;
      :DY = 30000.f ;
      :GRIDTYPE = "C" ;
      :DIFF_OPT = 1 ;
      :KM_OPT = 4 ;
      :DAMP_OPT = 0 ;
      ;DAMPCOEF = 0.2F ;
      :KHDIF = 0.f ;
      :KVDIF = 0.f ;
      :MP_PHYSICS = 3 ;
      :RA_LW_PHYSICS = 1 ;
      :RA_SW_PHYSICS = 1 ;
      :SF_SFCLAY_PHYSICS = 1 ;
     :SF_SURFACE_PHYSICS = 2 ;
      :BL_PBL_PHYSICS = 1 ;
      :CU_PHYSICS = 1 ;
```

```
 :SURFACE_INPUT_SOURCE = 1 ;
 :SST_UPDATE = 0 ;
 :GRID_FDDA = 1 ;
 :GFDDA_INTERVAL_M = 360 ;
 :GFDDA_END_H = 24 ;
 :SGFDDA_INTERVAL_M = 0 ;
 :SGFDDA_END_H = 0 ;
 :SF_URBAN_PHYSICS = 0 ;
 :FEEDBACK = 1 ;
 :SMOOTH_OPTION = 0 ;
 :SWRAD_SCAT = 1.f ;
 :W_DAMPING = 0 ;
 :MOIST_ADV_OPT = 1 ;
 :SCALAR_ADV_OPT = 1 ;
 :TKE_ADV_OPT = 1 ;
 :DIFF_6TH_OPT = 0 ;
 :DIFF_6TH_FACTOR = 0.12f ;
 :OBS_NUDGE_OPT = 0 ;
 :BUCKET_MM = -1.f ;
:BUCKET_J = -1.f ;
 :PREC_ACC_DT = 0.f ;
 :OMLCALL = 0 ;
 :ISFTCFLX = 0 ;
 ;ISHALLOW = 0 ;
 ;DFI_OPT = 0 ;
 ;SHCU_PHYSICS = 0 ;
 :WEST-EAST_PATCH_START_UNSTAG = 1 ;
 :WEST-EAST_PATCH_END_UNSTAG = 73 ;
 :WEST-EAST_PATCH_START_STAG = 1 ;
 :WEST-EAST_PATCH_END_STAG = 74 ;
 :SOUTH-NORTH_PATCH_START_UNSTAG = 1 ;
 :SOUTH-NORTH_PATCH_END_UNSTAG = 60 ;
 :SOUTH-NORTH_PATCH_START_STAG = 1 ;
 :SOUTH-NORTH_PATCH_END_STAG = 61 ;
 :BOTTOM-TOP_PATCH_START_UNSTAG = 1 ;
 :BOTTOM-TOP_PATCH_END_UNSTAG = 27 ;
 :BOTTOM-TOP_PATCH_START_STAG = 1 ;
 :BOTTOM-TOP_PATCH_END_STAG = 28 ;
 :GRID_ID = 1 ;
 :PARENT_ID = 0 ;
 :I_PARENT_START = 1 ;
 :J_PARENT_START = 1 ;
 :PARENT_GRID_RATIO = 1 ;
 :DT = 180.f ;
 :CEN_LAT = 34.83002f ;
 :CEN_LON = -81.03f ;
 :TRUELAT1 = 30.f ;
 :TRUELAT2 = 60.f ;
 :MOAD_CEN_LAT = 34.83002f ;
 :STAND_LON = -98.f ;
 :POLE_LAT = 90.f ;
 :POLE_LON = 0.f ;
 :GMT = 12.f ;
 :JULYR = 2000 ;
 :JULDAY = 24 ;
 :MAP_PROJ = 1 ;
 :MMINLU = "USGS" ;
 :NUM_LAND_CAT = 24 ;
 :ISWATER = 16 ;
 :ISLAKE = -1 ;
 :ISICE = 24 ;
 :ISURBAN = 1 ;
 :ISOILWATER = 14 ;
```

Special WRF Output Variables

The WRF model outputs the state variables defined in the Registry file, and these state variables are used in the model's prognostic equations. Some of these variables are perturbation fields; therefore the following definitions for reconstructing meteorological variables are necessary:

total geopotential	`PH + PHB`
total geopotential height in m	`( PH + PHB ) / 9.81`
total potential temperature in_ K	`T + 300`
total pressure in mb	`( P + PB ) * 0.01`
wind compoments, grid relative	`U, V`
surface pressure in Pa	`psfc`
surface winds, grid relative	`U10, V10 (valid at mass points)`
surface temperature and mixing ratio	`T2, Q2`

The definitions for map projection options:

`map_proj` = 1: Lambert Conformal
2: Polar Stereographic
3: Mercator
6: latitude and longitude (including global)