



BMKG

---

**METEOROLOGICAL TRAINING MODULE**

**01.06**

WEATHER RESEARCH AND FORECASTING MODEL  
INTRODUCTION AND APPLICATION

**WRF INSTALLATION  
PROCEDURE AND  
RUNNING PROCESS**

**Authors:**

**Dr. Danang Eko Nuryanto**

**Jaka Anugrah Ivanda Paski S.Tr**

**EDUCATION AND TRAINING CENTRE  
THE AGENCY FOR METEOROLOGY, CLIMATOLOGY AND GEOPHYSICS  
2020**

## FOREWORD

Praised be to God Almighty, teaching materials about of WRF Introduction: Model and Application has been completed. Basic knowledge and Description of WRF modelling system and application were provided.

This model is a next-generation mesoscale numerical weather prediction system designed for both atmospheric research and operational forecasting applications. It features two dynamical cores, a data assimilation system, and a software architecture supporting parallel computation and system extensibility. The model serves a wide range of meteorological applications across scales from tens of meters to thousands of kilometers.

The basic installation procedure and running processes was described in this module. The step by step of manual for installation and running processes with some examples also was provided in this module. A brief simulation procedure as an illustration of whether current architectural and engineering practice requiring careful consideration of local meteorology as a key factor also described.

Furthermore, input, criticism, and suggestions are needed for future improvements, so that these modules can be kept up-to-date and in tune with developments in science and technology.

Hopefully this teaching material can be useful and I express my gratitude and high appreciation to the drafting team for their contribution and cooperation.

Jakarta, October 2020

HEAD OF EDUCATION AND TRAINING CENTRE

THE AGENCY FOR METEOROLOGY CLIMATOLOGY AND GEOPHYSICS



Maman Sudarisman

## TABLE OF CONTENT

<b>FOREWORD .....</b>	<b>1</b>
<b>TABLE OF CONTENT.....</b>	<b>2</b>
<b>LIST OF FIGURES .....</b>	<b>4</b>
<b>CHAPTER I. PREFACE.....</b>	<b>5</b>
1.1. Introduction .....	5
1.2. Scope of Content.....	5
1.3. Tujuan Pembelajaran.....	<b>Error! Bookmark not defined.</b>
1.4. Contents .....	5
<b>CHAPTER II. INTRODUCTION AND APPLICATION.....</b>	<b>7</b>
2.1. WRF Modelling System .....	7
2.2. Two Dynamical Cores of WRF .....	7
2.3. WRF Parametrizations.....	8
2.4. WRF Application .....	11
2.5. WRF Types and Versions .....	12
2.6. WRF Modelling System Components .....	13
<b>BAB III. WRF MODEL (INSTALLATION PROCEDURE AND RUNNING PROCESS).....</b>	<b>18</b>
3.1. System Environment Tests .....	18
3.2. Library Compatibility Tests.....	24
3.3. Building WRF.....	26
3.4. Building WPS.....	34
3.5. Run WPS and WRF.....	41
<b>CHAPTER IV. SIMULATION PROCEDURES .....</b>	<b>46</b>
4.1. Domain Configuration .....	47
4.2. Input Data .....	47
4.3. Model Physics.....	47
4.4. Model Simulation .....	48

4.5. Postprocessing .....	48
<b>CHAPTER V. CLOSING .....</b>	<b>49</b>
5.1. Summary .....	49
5.2. Recommendation.....	49
<b>REFERENCES .....</b>	<b>50</b>

## LIST OF FIGURES

Figure 1. Schematic of the atmospheric boundary layer. Note that the presence of the surface layer, the mixed layer and a nocturnal stable layer. Also note the vertical scale of these atmospheric layers. From Stull (1988). .....	10
Figure 2. Flowchart for the WRF Modeling System Version 4.0 ( <a href="http://www2.mmm.ucar.edu/wrf/users/docs/user_guide_v4/v4.2/WRFUsersGuide_v42.pdf">http://www2.mmm.ucar.edu/wrf/users/docs/user_guide_v4/v4.2/WRFUsersGuide_v42.pdf</a> ).....	13

# CHAPTER I. PREFACE

## 1.1. Introduction

In this module, participants will learn about the basics of WRF Model and Application. You will walk away having learned the tools and techniques of WRF Model and Application to have basic technique to running the WRF project successfully. This course will discuss the basic knowledge for installation procedure and running process to be used as well as the most common simple techniques. This module will also cover some exercises that can be easily solved. This module is your first step toward learning how to complete various details of your project using the WRF model.

## 1.2. Scope of Content

The basic principles of WRF modeling and how the installation procedure is conducted, how the operation is carried out, and how the simulation and post-processing procedures are addressed in this training course

## 1.3. Learning Objectives

### 1.3.1. Goal

After following this lesson, participants are expected to have basic skills regarding WRF modeling techniques.

### 1.3.2. Objectives

- Participants are able to describe the important parts of the WRF running process.
- Participants are able to explain basic knowledge about what needs to be done before and after the WRF installation process.
- Participants are able to apply basic skills in developing WRF modeling and its application.

## 1.4. Contents

1. Introduction and Application
  - WRF Modelling System
  - Two Dynamical Cores of WRF
  - WRF Parametrization
  - WRF Application
  - WRF Types and Versions
  - WRF Modeling System Components

## 2. WRF Model (Installation Procedures and Running Process)

- System Environment Test
- Library Compatibility Test
- Building WRF
- Building WPS
- Run WPS and WRF

## 3. Simulation Procedures

- Domain Configuration
- Input Data
- Model Physics
- Model Simulation
- Post Processing

## CHAPTER II. INTRODUCTION AND APPLICATION

**Objective** : Participants are able to describe the important parts of the WRF running process.

### 2.1. WRF Modelling System

WRF is a mesoscale numerical weather prediction system designed for atmospheric research and operational forecasting applications. Model's development has begun in the late 1990s by the National Center for Atmospheric Research (NCAR), the National Oceanic and Atmospheric Administration (represented by the National Center for Environmental Prediction (NCEP) and Earth Systems Research Laboratory.), US Air Force, Laboratory Naval Research, University of Oklahoma, and Federal Aviation Administration (FAA). After years, this model has a large user community worldwide with a cumulative total of over 48,000 in 160 countries.

This model features two dynamic cores, a data assimilation system, and a software architecture that supports parallel computing and system extensibility. It serves meteorological applications at scales ranging from tens of meters to thousands of kilometers. WRF can produce simulations based on actual atmospheric conditions (i.e., observations and analyses) or ideal conditions. WRF can perform flexible and computationally efficient weather forecasting operations while reflecting the latest advances in physics, numerics, and data assimilation contributed by developers from the broad research community. This model is used operationally at NCEP and two other national meteorological centers.

<https://www.mmm.ucar.edu/weather-research-and-forecasting-model>

### 2.2. Two Dynamical Cores of WRF

The WRF system contains two dynamical solvers, referred to as the ARW (Advanced Research WRF) core and the NMM (Nonhydrostatic Mesoscale Model) core. The ARW has been developed in large part and is maintained by NCAR's Mesoscale and Microscale Meteorology Laboratory, and its users' page is: WRF-ARW Users' Page. <http://www2.mmm.ucar.edu/wrf/users/> The NMM core was developed by the National Centers for Environmental Prediction (NCEP), and is currently used in their HWRF (Hurricane WRF) system. <https://dtcenter.ucar.edu/wrf-nmm/users/>. The dynamical core includes most advection, pressure gradient, Coriolis, buoyancy, filter, diffusion, and time step, which is the dynamic core of Euler's mass with vertical coordinates following the terrain. ARW support and



development is centered at NCAR / MMM, and NMM development is centered at NCEP / EMC, and support is provided by NCAR / DTC (operationally only used for HWRF).

The ARW and NMM are suitable for use in a broad range of applications across scales ranging from meters to thousands of kilometers, including:

- Atmospheric physics/parameterization research
- Case-study research
- Real-time NWP and forecast system research
- Data assimilation research
- Teaching dynamics and NWP

ARW only

- Regional climate and seasonal time-scale research
- Coupled-chemistry applications
- Global simulations
- Idealized simulations at many scales (e.g. convection, baroclinic waves, large-eddy simulations)

### **2.3. WRF Parametrizations**

Treating physical processes is one of the most challenging aspects of modeling the climate system, known as parameterization. Atmospheric processes operate over an extensive range of time and space scales. However, because of computational cost, numerical integrations of the governing meteorological equations explicitly resolve only the primary energetic and phenomenological scales of motion. Nevertheless, there are significant interactions between the explicitly resolved large-scale flow and the truncated scales of motion, which must be accounted for somehow. Parameterization techniques seek to express these non-resolvable processes' statistical contribution to the time evolution of the explicitly resolved motions. Simultaneously, the contributions of these non-resolvable motions are diagnosed as functions of the large-scale fields.

The treatment of the water budget is the most difficult component of the parameterization problem. Convective-scale motions (i.e., motions on the order of several km) are responsible for most of the phase change and associated precipitation occurring in the atmosphere. These processes occur well below the resolvable scales of motion in a numerical model, but represent a very large, and often dominant, local energy source/sink in the climate system. Their effect on the time evolution of the large-scale motion fields are represented by what is most frequently referred to as a moist convective parameterization. Phase change

associated with these convective motions also plays an important role that must be represented in terms of the large-scale state variables. Examples of other physical processes that must be parameterized are the transfer of longwave and shortwave radiation, horizontal diffusion processes, and the effects of gravity waves. These are discussed further in following in the Earth's radiative energy budget through the formation of clouds.

Parameterization is necessary for several reasons:

- Computers are not yet powerful enough to treat many physical processes explicitly because they are either too small (as discussed earlier) or complex to be resolved;
- Many other physical processes cannot be explicitly modeled because they are not sufficiently understood to be represented in equation format or there are no appropriate data.

#### 2.3.1. Radiation transferred through the atmosphere.

Solar electromagnetic radiation is responsible for all dynamic and physical processes in the atmosphere, including cyclones, monsoons, tropical cyclones, and the Hadley circulation, to convection and coastal circulation at the mesoscale scale. Besides, the results of radiation processes such as fog, strong surface-based temperature inversions impact air quality, evaporation of water on the earth's surface, development of buoyancy instability leading to convective weather. Therefore, model simulations must represent radiation interaction with land, oceans, vegetation, clouds, air molecules, and mineral aerosols of natural and human origin.

#### 2.3.2. Planetary Boundary Layer and Surface Layer.

Atmospheric general circulation models need boundary conditions for the enthalpy, moisture (and momentum) equations: Fluxes of energy, water at the surface. The interaction of the atmosphere with the Earth's surface involves the exchange of heat, momentum, moisture and chemical species and hence is an important component of climate system modeling. Because the PBL is the physical "link" between the surface and the free atmosphere, it must be included in climate models.

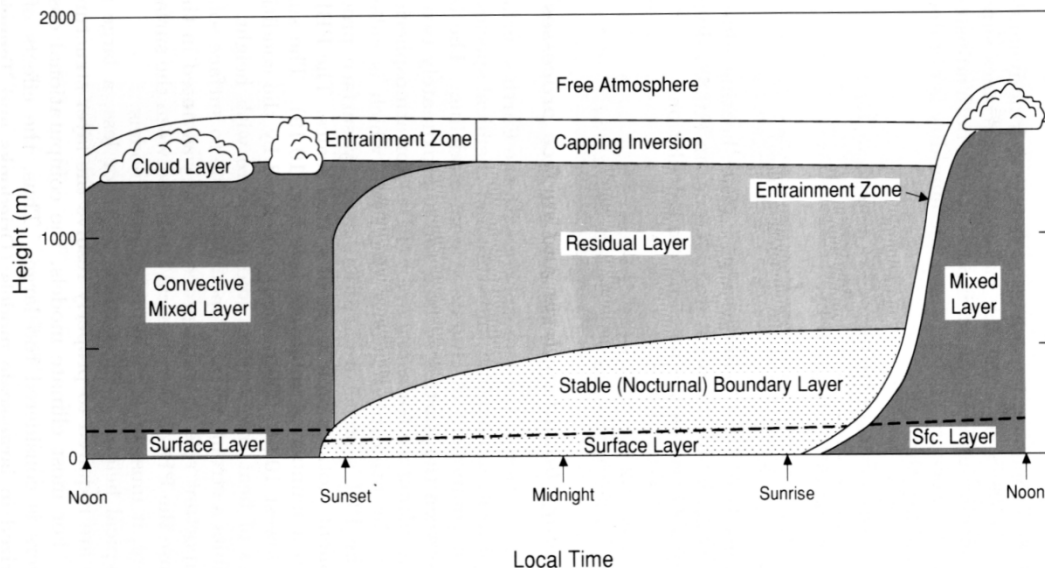


Figure 1. Schematic of the atmospheric boundary layer. Note that the presence of the surface layer, the mixed layer and a nocturnal stable layer. Also note the vertical scale of these atmospheric layers. From Stull (1988).

The surface layer is defined by the near vertical constancy (variations less than 10%) of the heat, momentum and moisture fluxes. For the surface layer, the momentum stresses are often based on bulk transfer coefficients for momentum, heat and moisture

#### 2.3.3. Cumulus Convection.

The effects of convective motions on the evolution of the thermal, moisture and even dynamical properties in the model atmosphere must be parameterized. Since the meteorological term for convective clouds is "cumulus", the problem is called cumulus parameterization in the atmospheric modeling community. Like any parameterization, the unresolved properties must be linked to the large-scale model variables. This "linking" process is known in the convective parameterization field as the "closure assumption".

#### 2.3.4. Microphysics of clouds and precipitation.

This is a central part of every model of the atmosphere. In numerical weather prediction they are important for quantitative precipitation forecasts. In climate modeling clouds are crucial due to their radiative impact, and aerosol-cloud-radiation effects are a major uncertainty in climate models. Cloud microphysical schemes have to describe the formation, growth and sedimentation of water particles (hydrometeors). They provide the latent heating rates for the dynamics. Cloud variables are treated by prognostic equations including advection.

## **2.4. WRF Application**

### **2.4.1. Parametrization research**

Parameterization is how we include the effects of physical processes implicitly when we cannot include the processes themselves explicitly. Parameterization can be thought of as emulation (modeling the effects of a process) rather than simulation (modeling the process itself).

[http://www2.mmm.ucar.edu/wrf/users/phys\\_references.html](http://www2.mmm.ucar.edu/wrf/users/phys_references.html)

### **2.4.2. Case studies**

WRF can overcome Earth system predictions outside of the weather and make two-way interactions with atmospheric components. These models are mostly compiled and run on WRF, not stand-alone models that run offline. For the air chemistry model, WRF-Chem is an in-line atmospheric chemistry model based on WRF. As for the Hydrological model, a WRF Hydrological Modeling Extension Package (WRF-Hydro) is available, which provides a fully integrated two-way interaction with WRF and stand-alone capabilities. In fire weather modeling, WRF-Fire is provided to help predict fire-behavior and the area of land burned. Several versions of WRF have been adapted to conditions such as HWRF, WRF-Urban, WRF-LES, and Polar WRF.

### **2.4.3. Short-range forecast**

The high-resolution domain configuration, the possible variety of input data, and the computational flexibility (especially in resource-constrained settings) mean that the WRF model is widely used in government centers and private companies worldwide. An example of using this model for short-term prediction is in the United States, NCEP uses WRF to support the National Weather Service in several systems such as Rapid Refresh (RAP) and High-Resolution Rapid Refresh (HRRR). NCEP also uses WRF in its short-range ensemble forecasting system (SREF). The ensemble model includes 13 ARW forecast members (on a 16-km grid) and its High-Resolution Window Forecast System (HIRESW), with a 4.2–3.5- km grid spacing domain for CONUS, Alaska, Hawaii, Puerto Rico, and Guam.

### **2.4.4. Data assimilation**

WRF Data Assimilation (WRFDA) is a specific WRF program to combine observation data with NWP product (the first guess or background forecast) and their respective error statistics to provide an improved estimate (the analysis) of the atmospheric state. Observations are used to make small corrections to a short-range forecast (background), which is assumed to be good, to produce a model analysis.

## **2.5. WRF Types and Versions**

The WRF modeling system has been in development for the past twenty years. The current release is Version 4, available since June 2018. The WRF model, specifically the ARW, is designed to be a flexible and sophisticated atmospheric simulation system that is portable and efficient on all available parallel computing platforms. A list of the versions of WRF that have been released is listed below, such as:

- Version 1.0: WRF was released December 2000
- Version 2.0: May 2004 (NMM added, EM nesting released)
- Version 2.1: August 2005 (EM becomes ARW)
- Version 2.2: December 2006 (WPS released)
- Version 3.0: April 2008 (includes global ARW version)
- Version 3.1: April 2009
- Version 3.2: April 2010
- Version 3.2.1: August 2010
- Version 3.3: April 2011
- Version 3.4: April 2012
- Version 3.5: January 2014
- Version 3.6: September 2014
- Version 3.7: April 2015
- Version 3.8: April 2016
- Version 3.8.1: August 2016
- Version 3.9: April 2017
- Version 3.9.1(.1): August 2017
- Version 4.0: June 2018

## 2.6. WRF Modelling System Components

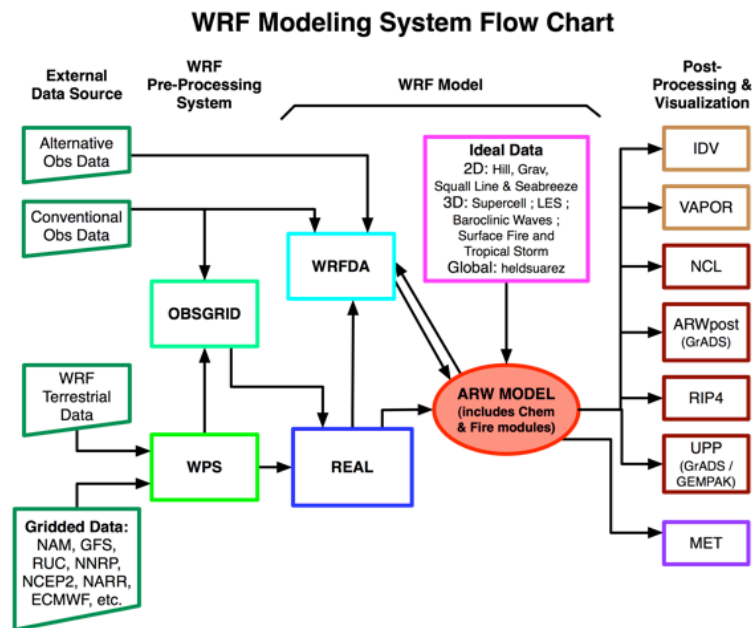


Figure 2. Flowchart for the WRF Modeling System Version 4.0  
([http://www2.mmm.ucar.edu/wrf/users/docs/user\\_guide\\_v4/v4.2/WRFUsersGuide\\_v42.pdf](http://www2.mmm.ucar.edu/wrf/users/docs/user_guide_v4/v4.2/WRFUsersGuide_v42.pdf))

As shown in the diagram, the WRF Modeling System consists of these major programs:

- The WRF Preprocessing System (WPS)
 

WRF Preprocessing System (WPS) is a combination of three primary programs that play a collective role in preparing input to the program for real data simulation. Each program performs a preparatory step like a geogrid used to define the model domain and interpolate static geographic data onto a grid. The ungrib program is used to extract the meteorological field from a GRIB file format. The metgrid program is required to interpolate the meteorological field extracted by the ungrib to a model grid defined by the geogrid. Meanwhile, the meteorological field's vertical interpolation to the WRF level is carried out in an adjustable program. This program is used primarily for real-data simulations. Its functions include

  - 1) Defining simulation domains;
  - 2) Interpolating terrestrial data (such as terrain, landuse, and soil types) to the simulation domain; and
  - 3) De-gribbing and interpolating meteorological data
- WRF-DA
 

This program is optional, but can be used to ingest observations into the interpolated analyses created by WPS. It can also be used to update WRF model's initial conditions

- ARW solver

This is the key component of the modeling system, which is composed of several initialization programs for idealized, and real-data simulations, and the numerical integration program.

- Post-processing & Visualization tools

Several programs are supported, including RIP4 (based on NCAR Graphics), NCAR Graphics Command Language (NCL), and conversion programs for other readily available graphics packages like GrADS.

- 1) Program VAPOR, Visualization and Analysis Platform for Ocean, Atmosphere, and Solar Researchers (<http://www.vapor.ucar.edu/>), is a 3-dimensional data visualization tool, and it is developed and supported by the VAPOR team at NCAR ([vapor@ucar.edu](mailto:vapor@ucar.edu)).
- 2) Program MET, Model Evaluation Tools (<http://www.dtcenter.org/met/users/>), is developed and supported by the Developmental Testbed Center at NCAR ([met\\_help@ucar.edu](mailto:met_help@ucar.edu)).

## 2.7. WRF Application Function (WPS, WRF RUN, ARW POST)

### 2.7.1. WPS

A brief description of each of the three main programs in WPS is given below, with further details presented in subsequent sections.

- Define simulation domain area (and nests)
- Produce terrain, landuse, soil type etc. on the simulation domain ("static" fields)
- De-grib GRIB files for meteorological data (u, v, T, q, surface pressure, soil data, snow data, sea-surface temperature, etc.)
- Interpolate meteorological data to WRF model grid (horizontally)
- Optionally add more observations to analysis (separate obsgrid program)

#### Program geogrid

This program aims to define domains and interpolate various terrestrial data sets into a grid model. The intended simulated domain can be determined using the user's WPS information in the "geogrid" rosters record of the WPS rosters file, namelist.wps. Many calculations are carried out by this program, such as latitude, longitude, and map scale factors at each grid point. The geogrid will interpolate soil categories, land use categories, terrain elevations, annual average soil temperature, monthly vegetation fraction, monthly albedo,

maximum albedo, snowfall, and slope. Global data for each of these fields has been provided on the WRF download page with resolutions of 30 ", 2 ', 5', and 10 ' ; here, " indicates the arc seconds and ' denotes the minute of arc. Apart from the default terrestrial plane interpolation, the geogrid program is general enough to interpolate most continuous fields and categories to the simulation domain.

In this program, new or additional data sets can be interpolated into the simulation domain by using a table file, GEOGRID.TBL. This file defines each field that the geogrid will generate and describes the interpolation method to use for a field, and the location on the file system where the data set for that field is located. The geogrid output is written in WRF I / O API format or NetCDF I / O format for easy visualization using external software packages, including NCVIEW, NCL, and RIP4.

### **Program ungrib**

The ungrib program reads the GRIB file, "dumps" the data, and writes the data in a simple format called an intermediate format. GRIB files contain time-varying data from the field of meteorology and usually come from other regional or global models, such as the NCEP NAM or GFS models. This program can read GRIB Issues 1 and 2, both versions of the GRIB format, using various codes to identify variables and levels in the GRIB file.

Ungrib uses a code table called Vtables to determine which columns to extract from the GRIB file and write to commonly provided intermediate formats such as for NAM 104 and 212 grids, AWIP NAM format, GFS, NCEP / NCAR Reanalysis archived in NCAR, RUC (pressure level data and hybrid WPS coordinate data), AGRMET AFWA ground-level model outputs, ECMWF, and other data sets. Additionally, users can create their Vtables.

This program can write intermediate data files in one of the three available formats, namely the new WPS format, which contains additional useful information for the downstream program, SI as the previous intermediate format of the WRF system, and the MM5 format.

### **Program metgrid**

Metgrid is a program that horizontally interpolates intermediate format meteorological data extracted by ungrib program to the geogrid program simulation domain. The WRF real program can then absorb the interpolated metgrid output. The METGRID.TBL file provides the configuration of this program for each field of meteorology. The output of the metgrid is written in the WRF I / O API format. Thus, by selecting the NetCDF I / O format, the metgrid can be made to write its output in NetCDF for easy visualization using an external software package and a geogrid output program.



### 2.7.2. WRF RUN

The WRF model is a fully compressible and nonhydrostatic model (with a run-time hydrostatic option). Its vertical coordinate is selectable as either a terrain-following (TF) or (beginning in Version 3.9) hybrid vertical coordinate (HVC) 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 gravitywave 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`), 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`).

Version 4 of the WRF model, 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
- Vertical refinement in a child domain

### 2.7.3. ARW POST

The ARWpost package reads-in WRF-ARW model data and creates GrADS output files. Since version 3.0 (released December 2010), vis5D output is no longer supported. More advanced 3D visualization tools, like VAPOR and IDV, have been developed over the last couple of years, and users are encouraged to explore those for their 3D visualization needs. The converter can read-in WPS geogrid and metgrid data, and WRF-ARW input and output files in netCDF format. Since version 3.0 the ARWpost code is no longer dependant on the WRF IO API. The advantage of this is that the ARWpost code can now be compiled and executed anywhere without the need to first install WRF. The disadvantage is that GRIB1 formatted WRF output files are no longer supported.

Obtain the ARWpost TAR file from the WRF Download page ([http://www2.mmm.ucar.edu/wrf/users/download/get\\_source.html](http://www2.mmm.ucar.edu/wrf/users/download/get_source.html)). Unzip and untar the ARWpost tar file. The tar file contains the following directories and files:

- README, a text file containing basic information on running ARWpost.
- arch/, directory containing configure and compilation control.
- clean, a script to clean compiled code.
- compile, a script to compile the code.
- configure, a script to configure the compilation for your system.
- namelist.ARWpost, namelist to control the running of the code.
- src/, directory containing all source code.
- scripts/, directory containing some grads sample scripts.
- util/, a directory containing some utilities.

### **Exercise**

1. What are the main features of the WRF model?
2. Why parameterization is so important in WRF modeling?
3. To process WRF model output, is ARWpost software always required?

## CHAPTER III. WRF MODEL (INSTALLATION PROCEDURE AND RUNNING PROCESS)

**Objective** : Participants are able to explain basic knowledge about what needs to be done before and after the WRF installation process

### 3.1. System Environment Tests

First and foremost, it is very important to have a gfortran compiler, as well as gcc and cpp. To test whether these exist on the system, type the following:

- o `which gfortran`
- o `which cpp`
- o `which gcc`

If you have these installed, you should be given a path for the location of each.

We recommend using a Fortran compiler that supports Fortran2003 standard (version 4.6 or later). To determine the version of gfortran you have, type:

```
gcc --version
```

Create a new, clean directory called Build\_WRF, and another one called TESTS.

There are a few simple tests that can be run to verify that the fortran compiler is built properly, and that it is compatible with the C compiler.

**NOTE:** If any of these tests fail, you will need to contact the systems administrator at your institution for help, as these are specific to your particular environment, and we do not have the resources to support these types of errors.

Below is a tar file that contains the tests. Download the tar file and place it in the TESTS directory.

[https://www2.mmm.ucar.edu/wrf/OnLineTutorial/compile\\_tutorial/tar\\_files/Fortran\\_C\\_tests.tar](https://www2.mmm.ucar.edu/wrf/OnLineTutorial/compile_tutorial/tar_files/Fortran_C_tests.tar)

To unpack the tar file, type:

```
tar -xf Fortran_C_tests.tar
```

There are 7 tests available, so start at the top and run through them, one at a time.

**Test #1: Fixed Format Fortran Test:** TEST\_1\_fortran\_only\_fixed.f

Type the following in the command line:

```
gfortran TEST_1_fortran_only_fixed.f
```

Now type:

```
./a.out
```

The following should print out to the screen:

```
SUCCESS test 1 fortran only fixed format
```

**Test #2: Free Format Fortran:** TEST\_2\_fortran\_only\_free.f90

Type the following in the command line:

```
gfortran TEST_2_fortran_only_free.f90
```

and then type:

```
./a.out
```

The following should print out to the screen:

```
Assume Fortran 2003: has FLUSH, ALLOCATABLE, derived type,  
and ISO C Binding  
SUCCESS test 2 fortran only free format
```

**Test #3: C:** TEST\_3\_c\_only.c

Type the following in the command line:

```
gcc TEST_3_c_only.c
```

and then type:

```
./a.out
```

The following should print out to the screen:

```
SUCCESS test 3 c only
```

Test #4: Fortran Calling a C Function (our gcc and gfortran have different defaults, so we force both to always use 64 bit [-m64] when combining them): TEST\_4\_fortran+c\_c.c, and TEST\_4\_fortran+x\_f.f90

Type the following in the command line:

```
gcc -c -m64 TEST_4_fortran+c_c.c
```

and then type:

```
gfortran -c -m64 TEST_4_fortran+c_f.f90
```

and then:

```
gfortran -m64 TEST_4_fortran+c_f.o TEST_4_fortran+c_c.o
```

and then issue:

```
./a.out
```

The following should print out to the screen:

```
C function called by Fortran
Values are xx = 2.00 and ii = 1
SUCCESS test 4 fortran calling c
```

In addition to the compilers required to manufacture the WRF executables, the WRF build system has scripts as the top level for the user interface. The WRF scripting system uses, and therefore having the following is necessary:

- o csh
- o perl
- o sh

To test whether these scripting languages are working properly on the system, there are 3 tests to run. These tests were included in the "Fortran and C Tests Tar File".

**Test #5:** csh In the command line, type:

```
./TEST_csh.csh
```

The result should be:

```
SUCCESS csh test
```

**Test #6:** perl In the command line, type:

```
./TEST_perl.pl
```

The result should be:

```
SUCCESS perl test
```

**Test #7:** sh In the command line, type:

```
./TEST_sh.sh
```

The result should be:

```
SUCCESS sh test
```

Finally, inside the scripts are quite a few UNIX commands that are available regardless of which shell is used. The following standard UNIX commands are mandatory:

ar	head	sed
awk	hostname	sleep
cat	ln	sort
cd	ls	tar
cp	make	touch
cut	mkdir	tr
expr	mv	uname
file	nm	wc
grep	printf	which
gzip	rm	m4

## Building Libraries

- Before getting started, you need to make another directory. Go inside your `Build_WRF` directory:

```
cd Build_WRF
```

and then make a directory called "LIBRARIES"

```
mkdir LIBRARIES
```

- Depending on the type of run you wish to make, there are various libraries that should be installed. Below are 5 libraries. Download all 5 tar files and place them in the `LIBRARIES` directory.

[mpich-3.0.4](#)

[https://www2.mmm.ucar.edu/wrf/OnLineTutorial/compile\\_tutorial/tar\\_files/mpich-3.0.4.tar.gz](https://www2.mmm.ucar.edu/wrf/OnLineTutorial/compile_tutorial/tar_files/mpich-3.0.4.tar.gz)

[netcdf-4.1.3](#)

[https://www2.mmm.ucar.edu/wrf/OnLineTutorial/compile\\_tutorial/tar\\_files/netcdf-4.1.3.tar.gz](https://www2.mmm.ucar.edu/wrf/OnLineTutorial/compile_tutorial/tar_files/netcdf-4.1.3.tar.gz)

[Jasper-1.900.1](#)

[https://www2.mmm.ucar.edu/wrf/OnLineTutorial/compile\\_tutorial/tar\\_files/jasper-1.900.1.tar.gz](https://www2.mmm.ucar.edu/wrf/OnLineTutorial/compile_tutorial/tar_files/jasper-1.900.1.tar.gz)

[libpng-1.2.50](#)

[https://www2.mmm.ucar.edu/wrf/OnLineTutorial/compile\\_tutorial/tar\\_files/libpng-1.2.50.tar.gz](https://www2.mmm.ucar.edu/wrf/OnLineTutorial/compile_tutorial/tar_files/libpng-1.2.50.tar.gz)

[zlib-1.2.7](#)

[https://www2.mmm.ucar.edu/wrf/OnLineTutorial/compile\\_tutorial/tar\\_files/zlib-1.2.7.tar.gz](https://www2.mmm.ucar.edu/wrf/OnLineTutorial/compile_tutorial/tar_files/zlib-1.2.7.tar.gz)

It is important to note that these libraries must all be installed with the same compilers as will be used to install WRF and WPS.

Before installing the libraries, these paths need to be set:

```
setenv DIR /path_to_directory/Build_WRF/LIBRARIES
setenv CC gcc
setenv CXX g++
setenv FC gfortran
setenv FCFLAGS -m64
setenv F77 gfortran
setenv FFLAGS -m64
setenv JASPERLIB $DIR/grib2/lib
setenv JASPERINC $DIR/grib2/include
setenv LDFLAGS -L$DIR/grib2/lib
setenv CPPFLAGS -I$DIR/grib2/include
```

1. **NetCDF**: This library is always necessary!

```
tar xzvf netcdf-4.1.3.tar.gz      #or just .tar if no .gz
present
cd netcdf-4.1.3
./configure --prefix=$DIR/netcdf --disable-dap
        --disable-netcdf-4 --disable-shared
make
make install
setenv PATH $DIR/netcdf/bin:$PATH
setenv NETCDF $DIR/netcdf
cd ..
```

2. **MPICH**: This library is necessary if you are planning to build WRF in parallel. If your machine does not have more than 1 processor, or if you have no need to run WRF with multiple processors, you can skip installing MPICH.

In principle, any implementation of the MPI-2 standard should work with WRF; however, we have the most experience with MPICH, and therefore, that is what will be described here.

Assuming all the 'setenv' commands were already issued while setting up NetCDF, you can continue on to install MPICH, issuing each of the following commands:

```
tar xzvf mpich-3.0.4.tar.gz      #or just .tar if no .gz
present
cd mpich-3.0.4
./configure --prefix=$DIR/mpich
make
make install
setenv PATH $DIR/mpich/bin:$PATH
cd ..
```

3. **zlib**: This is a compression library necessary for compiling WPS (specifically ungrib) with GRIB2 capability.

Assuming all the "setenv" commands from the NetCDF install are already set, you can move on to the commands to install zlib.



```
tar xzvf zlib-1.2.7.tar.gz      #or just .tar if no .gz present
cd zlib-1.2.7
./configure --prefix=$DIR/grib2
make
make install
cd ..
```

4. **libpng**: This is a compression library necessary for compiling WPS (specifically ungrib) with GRIB2 capability.

Assuming all the "setenv" commands from the NetCDF install are already set, you can move on to the commands to install zlib.

```
tar xzvf libpng-1.2.50.tar.gz   #or just .tar if no .gz present
cd libpng-1.2.50
./configure --prefix=$DIR/grib2
make
make install
cd ..
```

5. **JasPer**: This is a compression library necessary for compiling WPS (specifically ungrib) with GRIB2 capability.

Assuming all the "setenv" commands from the NetCDF install are already set, you can move on to the commands to install zlib.

```
tar xzvf jasper-1.900.1.tar.gz  #or just .tar if no .gz present
cd jasper-1.900.1
./configure --prefix=$DIR/grib2
make
make install
cd ..
```

### 3.2. Library Compatibility Tests

Once the target machine is able to make small Fortran and C executables (what was verified in the System Environment Tests section), and after the NetCDF and MPI libraries are constructed (two of the libraries from the Building Libraries section), to emulate the WRF code's behavior, two additional small tests are required. We need to verify that the libraries

are able to work with the compilers that are to be used for the WPS and WRF builds.

NOTE: If any of these tests fail, you will need to contact the systems administrator at your institution for help, as these are specific to your particular environment, and we do not have the resources to support these types of errors.

Below is a tar file that contains these tests. Download this tar file and place it in the TESTS directory, and then "cd" into the TESTS directory:

[https://www2.mmm.ucar.edu/wrf/OnLineTutorial/compile\\_tutorial/tar\\_files/Fortran\\_C\\_NETCDF\\_MPI\\_tests.tar](https://www2.mmm.ucar.edu/wrf/OnLineTutorial/compile_tutorial/tar_files/Fortran_C_NETCDF_MPI_tests.tar)

To unpack the tar file, type:

```
tar -xf Fortran_C_NETCDF_MPI_tests.tar
```

There are 2 tests:

#### 1. Test #1: Fortran + C + NetCDF

The NetCDF-only test requires the include file from the NETCDF package be in this directory. Copy the file here:

```
cp ${NETCDF}/include/netcdf.inc .
```

Compile the Fortran and C codes for the purpose of this test (the -c option says to not try to build an executable). Type the following commands:

```
gfortran -c 01_fortran+c+netcdf_f.f
gcc -c 01_fortran+c+netcdf_c.c
gfortran 01_fortran+c+netcdf_f.o 01_fortran+c+netcdf_c.o \
    -L${NETCDF}/lib -lnetcdff -lnetcdf
./a.out
```

The following should be displayed on your screen:

```
C function called by Fortran
Values are xx = 2.00 and ii = 1
SUCCESS test 1 fortran + c + netcdf
```

#### **Test #2:** Fortran + C + NetCDF + MPI

The NetCDF+MPI test requires include files from both of these packages be in this directory,

but the MPI scripts automatically make the `mpif.h` file available without assistance, so no need to copy that one. Copy the NetCDF include file here:

```
cp ${NETCDF}/include/netcdf.inc .
```

Note that the MPI executables `mpif90` and `mpicc` are used below when compiling. Issue the following commands:

```
mpif90 -c 02_fortran+c+netcdf+mpi_f.f
mpicc -c 02_fortran+c+netcdf+mpi_c.c
mpif90 02_fortran+c+netcdf+mpi_f.o \
02_fortran+c+netcdf+mpi_c.o \
-L${NETCDF}/lib -lnetcdff -lnetcdf
mpirun ./a.out
```

The following should be displayed on your screen:

```
C function called by Fortran
Values are xx = 2.00 and ii = 1
status = 2
SUCCESS test 2 fortran + c + netcdf + mpi
```

### 3.3. Building WRF

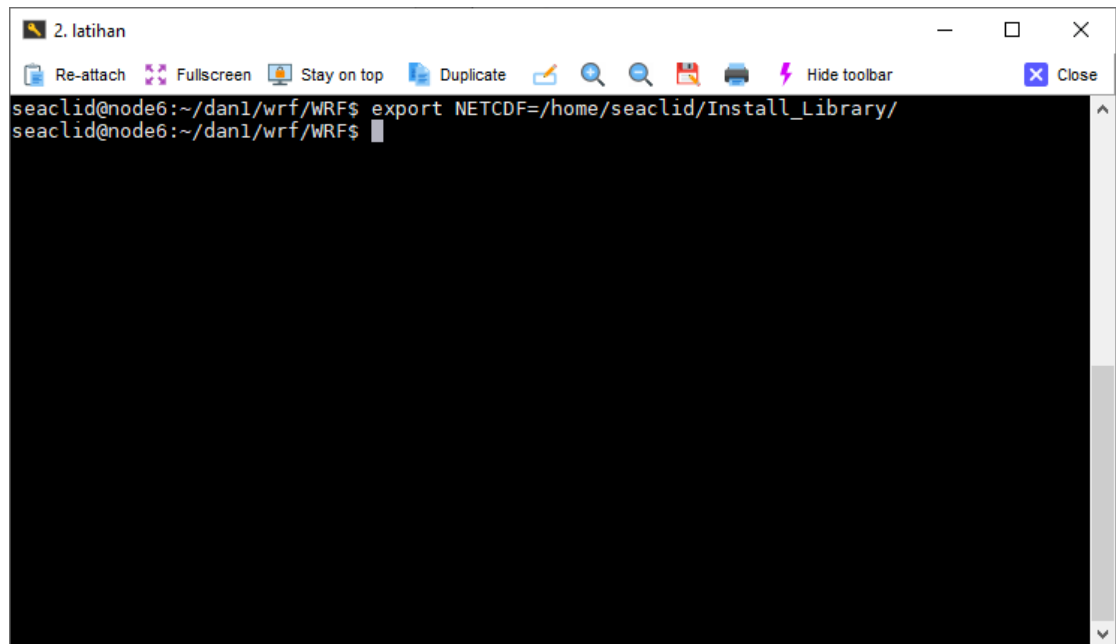
1. After ensuring that all libraries are compatible with the compilers, you can now prepare to build WRF. You can obtain the WRF source code by following the steps beginning with [http://www2.mmm.ucar.edu/wrf/users/download/get\\_source.html](http://www2.mmm.ucar.edu/wrf/users/download/get_source.html).

Once you obtain the WRF source code (and if you downloaded a tar file, have unpacked the tar file), go into the WRF directory:

```
cd WRF
```

Set the path for environment of NETCDF library

```
export NETCDF = <path-of-netcdf-library>
```

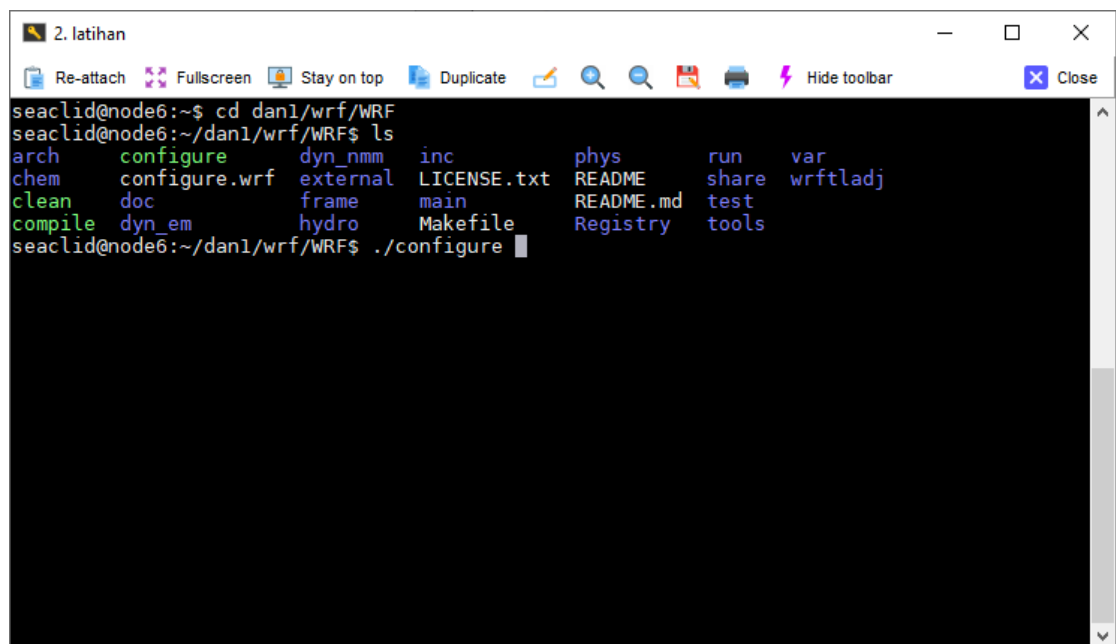


A terminal window titled "2. latihan" with a toolbar containing icons for Re-attach, Fullscreen, Stay on top, Duplicate, and other functions. The terminal shows the user "seaclid" at "node6" in the directory "~/dan1/wrf/WRF". The command "export NETCDF=/home/seaclid/Install\_Library/" has been entered and executed, resulting in a new prompt.

```
seaclid@node6:~/dan1/wrf/WRF$ export NETCDF=/home/seaclid/Install_Library/
seaclid@node6:~/dan1/wrf/WRF$
```

2. Create a configuration file for your computer and compiler:

```
./configure
```



A terminal window titled "2. latihan" showing the directory contents of "~/dan1/wrf/WRF" after running "ls". The output lists various subdirectories and files. The command "cd dan1/wrf/WRF" has been entered, and the prompt has changed to "seaclid@node6:~/dan1/wrf/WRF\$". The command "ls" has been executed, showing the following output:

```
seaclid@node6:~$ cd dan1/wrf/WRF
seaclid@node6:~/dan1/wrf/WRF$ ls
arch      configure  dyn_nmm  inc      phys      run      var
chem      configure.wrf  external LICENSE.txt  README  share  wrftldj
clean     doc        frame    main     README.md test
compile  dyn_em     hydro    Makefile Registry  tools
seaclid@node6:~/dan1/wrf/WRF$ ./configure
```

```

2. latihan
Re-attach Fullscreen Stay on top Duplicate Hide toolbar Close

seaclid@node6:~/dan1/wrf/WRF$ ./configure
checking for perl5... no
checking for perl... found /usr/bin/perl (perl)
Will use NETCDF in dir: /home/seaclid/Install_Library/
HDF5 not set in environment. Will configure WRF for use without.
PHDF5 not set in environment. Will configure WRF for use without.
Will use 'time' to report timing information
$JASPERLIB or $JASPERINC not found in environment, configuring to build without grib2 I/O
...
-----
Please select from among the following Linux x86_64 options:

  1. (serial)  2. (smpar)  3. (dmpar)  4. (dm+sm)  PGI (pgf90/gcc)
  5. (serial)  6. (smpar)  7. (dmpar)  8. (dm+sm)  PGI (pgf90/pgcc): SGI MPT
  9. (serial) 10. (smpar) 11. (dmpar) 12. (dm+sm)  PGI (pgf90/gcc): PGI accelerator
 13. (serial) 14. (smpar) 15. (dmpar) 16. (dm+sm)  INTEL (ifort/icc)
                                     17. (dm+sm)  INTEL (ifort/icc): Xeon Phi (MIC a
rchitecture)
 18. (serial) 19. (smpar) 20. (dmpar) 21. (dm+sm)  INTEL (ifort/icc): Xeon (SNB with
AVX mods)
 22. (serial) 23. (smpar) 24. (dmpar) 25. (dm+sm)  INTEL (ifort/icc): SGI MPT
 26. (serial) 27. (smpar) 28. (dmpar) 29. (dm+sm)  INTEL (ifort/icc): IBM POE
 30. (serial) 31. (dmpar) 32. (serial) 33. (smpar) 34. (dmpar) 35. (dm+sm)  PATHSCALE (pathf90/pathcc)
 36. (serial) 37. (smpar) 38. (dmpar) 39. (dm+sm)  GNU (gfortran/gcc)
 40. (serial) 41. (smpar) 42. (dmpar) 43. (dm+sm)  IBM (xlf90_r/cc_r)
 44. (serial) 45. (smpar) 46. (dmpar) 47. (dm+sm)  PGI (ftn/gcc): Cray XC CLE
                                     CRAY CCE (ftn $(NOOMP)/cc): Cray X
E and XC
 48. (serial) 49. (smpar) 50. (dmpar) 51. (dm+sm)  INTEL (ftn/icc): Cray XC
 52. (serial) 53. (smpar) 54. (dmpar) 55. (dm+sm)  PGI (pgf90/pgcc)
 56. (serial) 57. (smpar) 58. (dmpar) 59. (dm+sm)  PGI (pgf90/gcc): -f90=pgf90
 60. (serial) 61. (smpar) 62. (dmpar) 63. (dm+sm)  PGI (pgf90/pgcc): -f90=pgf90
 64. (serial) 65. (smpar) 66. (dmpar) 67. (dm+sm)  INTEL (ifort/icc): HSW/BDW
 68. (serial) 69. (smpar) 70. (dmpar) 71. (dm+sm)  INTEL (ifort/icc): KNL MIC
 72. (serial) 73. (smpar) 74. (dmpar) 75. (dm+sm)  FUJITSU (frtpx/fccpx): FX10/FX100
SPARC64 IXfx/Xlfx

Enter selection [1-75] : 

```

You will see various options. Choose the option that lists the compiler you are using and the way you wish to build WRF (i.e., serially or in parallel). Although there are 3 different types of parallel (smpar=*shared memory parallel*, dmpar=*distributed memory parallel*, and dm+sm=*both*), we have the most experience with dmpar and typically recommend choosing this option.

```

2. latihan
Re-attach Fullscreen Stay on top Duplicate Hide toolbar Close

seaclid@node6:~/dan1/wrf/WRF$ ./configure
checking for perl5... no
checking for perl... found /usr/bin/perl (perl)
Will use NETCDF in dir: /home/seaclid/Install/Library/
HDF5 not set in environment. Will configure WRF for use without.
PHDF5 not set in environment. Will configure WRF for use without.
Will use 'time' to report timing information
$JASPERLIB or $JASPERINC not found in environment, configuring to build without grib2 I/O
...
Please select from among the following Linux x86_64 options:

1. (serial) 2. (smpar) 3. (dmpar) 4. (dm+sm) PGI (pgf90/gcc)
5. (serial) 6. (smpar) 7. (dmpar) 8. (dm+sm) PGI (pgf90/pgcc): SGI MPT
9. (serial) 10. (smpar) 11. (dmpar) 12. (dm+sm) PGI (pgf90/gcc): PGI accelerator
13. (serial) 14. (smpar) 15. (dmpar) 16. (dm+sm) INTEL (ifort/icc)
17. (dm+sm) INTEL (ifort/icc): Xeon Phi (MIC a
rchitecture)
18. (serial) 19. (smpar) 20. (dmpar) 21. (dm+sm) INTEL (ifort/icc): Xeon (SNB with
AVX mods)
22. (serial) 23. (smpar) 24. (dmpar) 25. (dm+sm) INTEL (ifort/icc): SGI MPT
26. (serial) 27. (smpar) 28. (dmpar) 29. (dm+sm) INTEL (ifort/icc): IBM POE
30. (serial) 31. (dmpar) PATHSCALE (pathf90/pathcc)
32. (serial) 33. (smpar) 34. (dmpar) 35. (dm+sm) GNU (gfortran/gcc)
36. (serial) 37. (smpar) 38. (dmpar) 39. (dm+sm) IBM (xlf90_r/cc_r)
40. (serial) 41. (smpar) 42. (dmpar) 43. (dm+sm) PGI (ftn/gcc): Cray XC CLE
44. (serial) 45. (smpar) 46. (dmpar) 47. (dm+sm) CRAY CCE (ftn $(NOOMP)/cc): Cray X
E and XC
48. (serial) 49. (smpar) 50. (dmpar) 51. (dm+sm) INTEL (ftn/icc): Cray XC
52. (serial) 53. (smpar) 54. (dmpar) 55. (dm+sm) PGI (pgf90/pgcc)
56. (serial) 57. (smpar) 58. (dmpar) 59. (dm+sm) PGI (pgf90/gcc): -f90=pgf90
60. (serial) 61. (smpar) 62. (dmpar) 63. (dm+sm) PGI (pgf90/pgcc): -f90=pgf90
64. (serial) 65. (smpar) 66. (dmpar) 67. (dm+sm) INTEL (ifort/icc): HSW/BDW
68. (serial) 69. (smpar) 70. (dmpar) 71. (dm+sm) INTEL (ifort/icc): KNL MIC
72. (serial) 73. (smpar) 74. (dmpar) 75. (dm+sm) FUJITSU (frtpr/fccpx): FX10/FX100
SPARC64 IXfx/Xlfx

Enter selection [1-75] : 34

```

```

2. latihan
Re-attach Fullscreen Stay on top Duplicate Hide toolbar Close

60. (serial) 61. (smpar) 62. (dmpar) 63. (dm+sm) PGI (pgf90/pgcc): -f90=pgf90
64. (serial) 65. (smpar) 66. (dmpar) 67. (dm+sm) INTEL (ifort/icc): HSW/BDW
68. (serial) 69. (smpar) 70. (dmpar) 71. (dm+sm) INTEL (ifort/icc): KNL MIC
72. (serial) 73. (smpar) 74. (dmpar) 75. (dm+sm) FUJITSU (frtpr/fccpx): FX10/FX100
SPARC64 IXfx/Xlfx

Enter selection [1-75] : 34

Compile for nesting? (1=basic, 2=preset moves, 3=vortex following) [default 1]: 1

Configuration successful!

testing for fseeke and fseeke64
fseeke64 is supported

# Settings for Linux x86_64 ppc64le, gfortran compiler with gcc (dmpar)
#
DESCRIPTION = GNU ($SFC/$SCC)
DMPARALLEL = 1
OMPCPP = # -D_OPENMP
OMP = # -fopenmp
OMPCC = # -fopenmp
SFC = gfortran
SCC = gcc
CCOMP = gcc
DM_FC = mpif90 -f90=$(SFC)
DM_CC = mpicc -cc=$(SCC)
FC = time $(DM_FC)
CC = $(DM_CC) -DFSEEK064_OK
LD = $(FC)
RWORDSIZE = $(NATIVE_RWORDSIZE)
PROMOTION = #-fdefault-real-8
ARCH_LOCAL = -DNONSTANDARD_SYSTEM_SUBR -DWRF_USE_CLM $(NETCDF4_IO_OPTS)
CFLAGS_LOCAL = -w -O3 -c
LDFLAGS_LOCAL =
CPLUSPLUSLIB =
ESMF_LDFLAG = $(CPLUSPLUSLIB)
FCOPTIM = -O2 -ftree-vectorize -funroll-loops
FCREDUCEDOPT = $(FCOPTIM)

```

```

2. latihan
Re-attach Fullscreen Stay on top Duplicate Hide toolbar Close

ENVCOMPDEFS =
WRF_CHEM = 0
CPPFLAGS = $(ARCHFLAGS) $(ENVCOMPDEFS) -I$(LIBINCLUDE) $(TRADFLAG)
NETCDFPATH = /home/seaclid/Install_Library
HDF5PATH =
WRFPLUSPATH =
RTTOVPATH =
PNETCDFPATH =

bundled: io_only
external: io_only $(WRF_SRC_ROOT_DIR)/external/RSL_LITE/librsl_lite.a gen_comms_rslite module_dm_rslite $(ESMF_TARGET)
io_only: esmf time wrfio_nf \
        wrf_ioapi_includes wrfio_grib_share wrfio_grib1 wrfio_int fftpack

#####
-----
Settings listed above are written to configure.wrf.
If you wish to change settings, please edit that file.
If you wish to change the default options, edit the file:
    arch/configure.defaults
NetCDF users note:
This installation of NetCDF supports large file support. To DISABLE large file
support in NetCDF, set the environment variable WRFIO_NCD_NO_LARGE_FILE_SUPPORT
to 1 and run configure again. Set to any other value to avoid this message.

Testing for NetCDF, C and Fortran compiler

This installation of NetCDF is 64-bit
        C compiler is 64-bit
        Fortran compiler is 64-bit
        It will build in 64-bit

*****
This build of WRF will use NETCDF4 with HDF5 compression
*****

seaclid@node6:~/dan1/wrf/WRF$

```

- Once your configuration is complete, you should have a `configure.wrf` file, and you are ready to compile. To compile WRF, you will need to decide which type of case you wish to compile. The options are listed below:

```

em_real (3d real case)
em_quarter_ss (3d ideal case)
em_b_wave (3d ideal case)
em_les (3d ideal case)
em_heldsuarez (3d ideal case)
em_tropical_cyclone (3d ideal case)
em_hill2d_x (2d ideal case)
em_squall2d_x (2d ideal case)
em_squall2d_y (2d ideal case)
em_grav2d_x (2d ideal case)

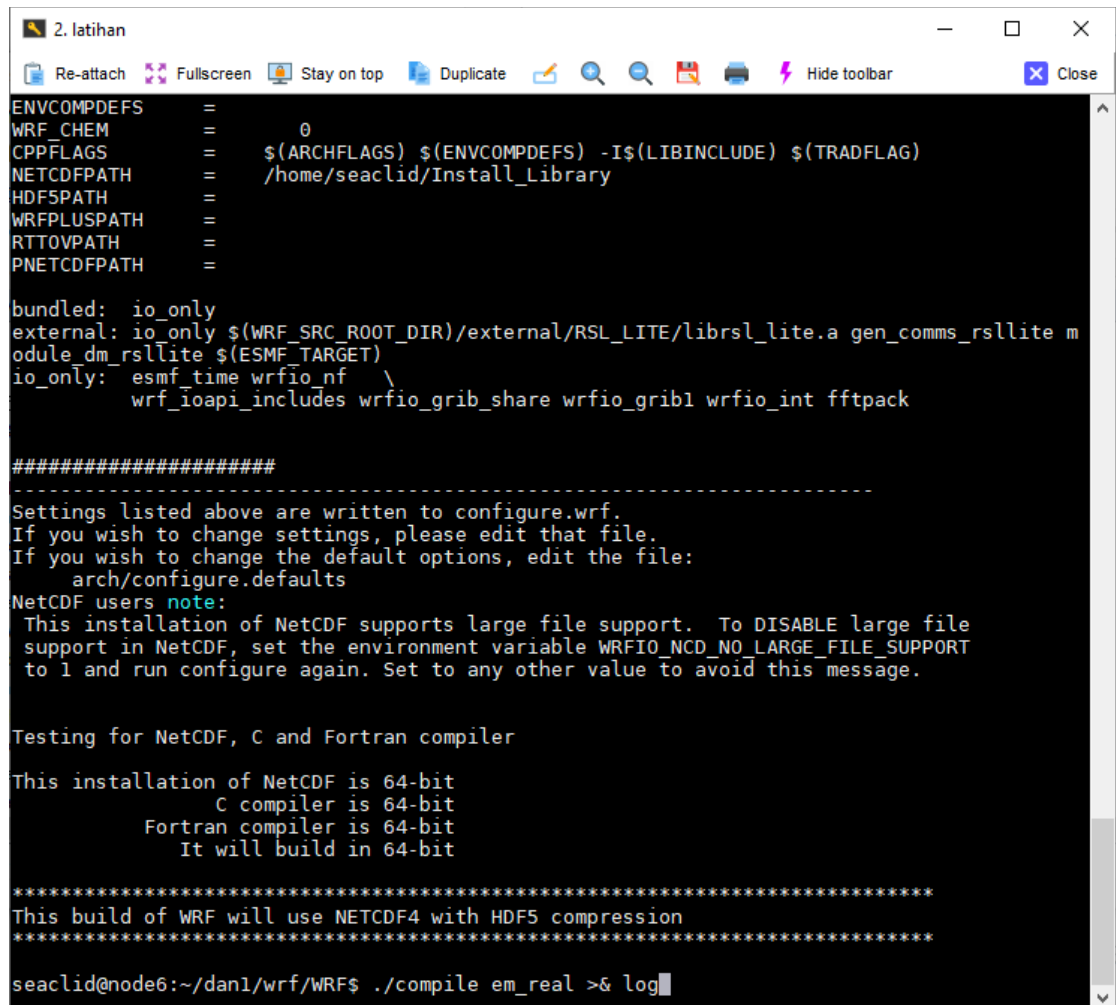
```

em\_seabreeze2d\_x (2d ideal case)

em\_scm\_xy (1d ideal case)

```
./compile case_name >& log.compile
```

where `case_name` is one of the options listed above



```
2. latihan
Re-attach Fullscreen Stay on top Duplicate
ENVCOMPDEFS =
WRF_CHEM = 0
CPPFLAGS = $(ARCHFLAGS) $(ENVCOMPDEFS) -I$(LIBINCLUDE) $(TRADFLAG)
NETCDFPATH = /home/seaclid/Install_Library
HDF5PATH =
WRFPLUSPATH =
RTTOVPATH =
PNETCDFPATH =

bundled: io_only
external: io_only $(WRF_SRC_ROOT_DIR)/external/RSL_LITE/librsl_lite.a gen_comms_rslite m
odule_dm_rslite $(ESMF_TARGET)
io_only: esmf_time wrfio_nf \
        wrf_ioapi_includes wrfio_grib_share wrfio_grib1 wrfio_int fftpack

#####
-----
Settings listed above are written to configure.wrf.
If you wish to change settings, please edit that file.
If you wish to change the default options, edit the file:
    arch/configure.defaults
NetCDF users note:
  This installation of NetCDF supports large file support. To DISABLE large file
  support in NetCDF, set the environment variable WRFIO_NCD_NO_LARGE_FILE_SUPPORT
  to 1 and run configure again. Set to any other value to avoid this message.

Testing for NetCDF, C and Fortran compiler

This installation of NetCDF is 64-bit
      C compiler is 64-bit
      Fortran compiler is 64-bit
      It will build in 64-bit

*****
This build of WRF will use NETCDF4 with HDF5 compression
*****

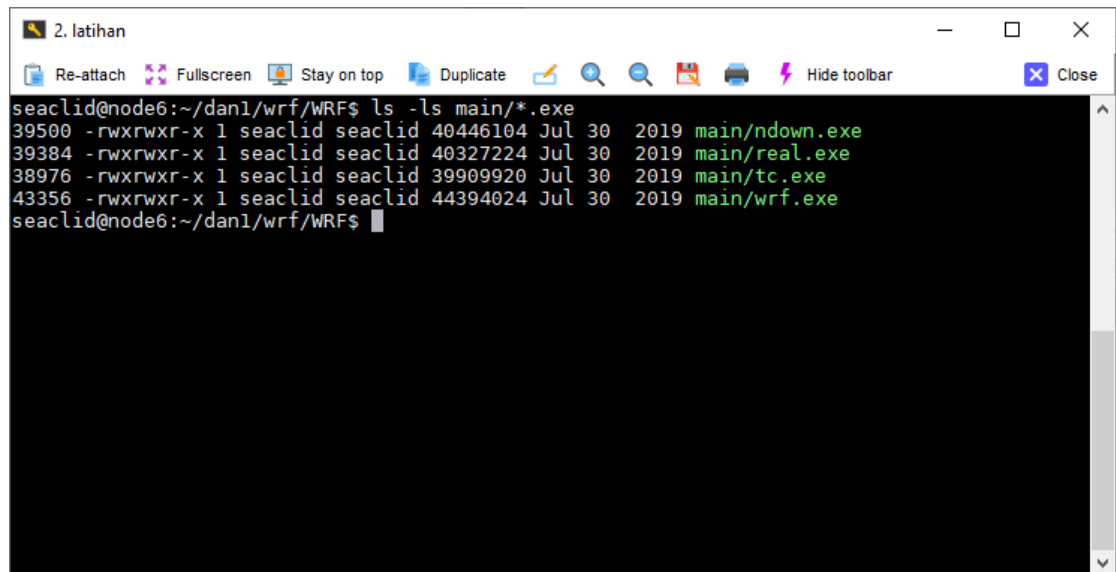
seaclid@node6:~/dan1/wrf/WRF$ ./compile em_real >& log
```

Compilation should take about 45-60 minutes.

4. Once the compilation completes, to check whether it was successful, you need to look for executables in the `WRF/main` directory:

```
ls -ls main/*.exe
```





```
2. latihan
Re-attach Fullscreen Stay on top Duplicate
seaclid@node6:~/dan1/wrf/WRF$ ls -ls main/*.exe
39500 -rwxrwxr-x 1 seaclid seaclid 40446104 Jul 30 2019 main/ndown.exe
39384 -rwxrwxr-x 1 seaclid seaclid 40327224 Jul 30 2019 main/real.exe
38976 -rwxrwxr-x 1 seaclid seaclid 39909920 Jul 30 2019 main/tc.exe
43356 -rwxrwxr-x 1 seaclid seaclid 44394024 Jul 30 2019 main/wrf.exe
seaclid@node6:~/dan1/wrf/WRF$
```

If you compiled a real case, you should see:

- wrf.exe (model executable)
- real.exe (real data initialization)
- ndown.exe (one-way nesting)
- tc.exe (for tc bogusing--serial only)

If you compiled an idealized case, you should see:

- wrf.exe (model executable)
- ideal.exe (ideal case initialization)

These executables are linked to 2 different directories:

- WRF/run
- WRF/test/em\_real

```
2. latihan
Re-attach Fullscreen Stay on top Duplicate Hide toolbar Close
seaclid@node6:~/dan1/wrf/WRF$ ls run
aerosol.formatted          ishmael-gamma-tab.bin
aerosol_lat.formatted       ishmael-qi-qc.bin
aerosol_lon.formatted       ishmael-qi-qr.bin
aerosol_plev.formatted      kernels.asc_s_0_03_0_9
BROADBAND_CLOUD_GODDARD.bin kernels_z.asc
bulkdens.asc_s_0_03_0_9    LANDUSE.TBL
bulkradii.asc_s_0_03_0_9   masses.asc
CAM_ABS_DATA               MPTABLE.TBL
CAM_AEROPT_DATA            namelist.input
CAMtr_volume_mixing_ratio.A1B ndown.exe
CAMtr_volume_mixing_ratio.A2 ozone.formatted
CAMtr_volume_mixing_ratio.RCP4.5 ozone_lat.formatted
CAMtr_volume_mixing_ratio.RCP6 ozone_plev.formatted
CAMtr_volume_mixing_ratio.RCP8.5 p3_lookup_table_1.dat-v4.1
capacity.asc               p3_lookup_table_2.dat-v4.1
CCN_ACTIVATION.BIN         README.namelist
CLM_ALB_ICE_DFS_DATA       README.rasm_diag
CLM_ALB_ICE_DRC_DATA       README.tslist
CLM_ASM_ICE_DFS_DATA       real.exe
CLM_ASM_ICE_DRC_DATA       RRTM_DATA
CLM_DRDSDT0_DATA          RRTM_DATA_DBL
CLM_EXT_ICE_DFS_DATA       RRTMG_LW_DATA
CLM_EXT_ICE_DRC_DATA       RRTMG_LW_DATA_DBL
CLM_KAPPA_DATA            RRTMG_SW_DATA
CLM_TAU_DATA              RRTMG_SW_DATA_DBL
co2_trans                 SOILPARM.TBL
coeff_p.asc               SOILPARM.TBL_Kishne_2017
coeff_q.asc               tc.exe
constants.asc             termvels.asc
create_p3_lookupTable_1.f90 tr49t67
ETAMPNEW_DATA             tr49t85
ETAMPNEW_DATA_DBL         tr67t85
ETAMPNEW_DATA.expanded_rain URBPARM.TBL
ETAMPNEW_DATA.expanded_rain_DBL URBPARM_UZE.TBL
GENPARM.TBL              VEGPARM.TBL
grib2map.tbl             wind-turbine-1.tbl
gribmap.txt              wrf.exe
HLC.TBL
seaclid@node6:~/dan1/wrf/WRF$
```

```

2. latihan
Re-attach Fullscreen Stay on top Duplicate
seaclid@node6:~/dan1/wrf/WRF$ ls test/em_real
aerosol.formatted          MPTABLE.TBL
aerosol_lat.formatted      namelist.input
aerosol_lon.formatted      namelist.input.4km
aerosol_plev.formatted     namelist.input.chem
BROADBAND_CLOUD_GODDARD.bin namelist.input.diaqs
bulkdens.asc_s_0_03_0_9   namelist.input.fire
bulkradii.asc_s_0_03_0_9  namelist.input.global
CAM_ABS_DATA              namelist.input.jan00
CAM_AEROPT_DATA           namelist.input.jun01
CAMtr_volume_mixing_ratio namelist.input.ndown_1
CAMtr_volume_mixing_ratio.A1B namelist.input.ndown_2
CAMtr_volume_mixing_ratio.A2 namelist.input.ndown_3
CAMtr_volume_mixing_ratio.RCP4.5 namelist.input.pbl-les
CAMtr_volume_mixing_ratio.RCP6 namelist.input.volc
capacity.asc              ndown.exe
CCN_ACTIVATE.BIN          ozone.formatted
CLM_ALB_ICE_DFS_DATA      ozone_lat.formatted
CLM_ALB_ICE_DRC_DATA      ozone_plev.formatted
CLM_ASM_ICE_DFS_DATA      p3_lookup_table_1.dat-v4.1
CLM_ASM_ICE_DRC_DATA      p3_lookup_table_2.dat-v4.1
CLM_DRDSDT0_DATA          README.grid_fdda
CLM_EXT_ICE_DFS_DATA      README.namelist
CLM_EXT_ICE_DRC_DATA      README.obs_fdda
CLM_KAPPA_DATA            real.exe
CLM_TAU_DATA              RRTM_DATA
coeff_p.asc               RRTMG_LW_DATA
coeff_q.asc               RRTMG_SW_DATA
constants.asc             run_1way.tar
ETAMPNEW_DATA             run_2way.tar
ETAMPNEW_DATA.expanded_rain run_restart.tar
examples.namelist         sample.txt
GENPARM.TBL               SOILPARM.TBL
grib2map.tbl              tc.exe
gribmap.txt               termvels.asc
HLC.TBL                   tr49t67
ishmael-gamma-tab.bin     tr49t85
ishmael-qi-qc.bin         tr67t85
ishmael-qi-qr.bin         URBPARM.TBL
kernels.asc_s_0_03_0_9    VEGPARM.TBL
kernels_z.asc             wind-turbine-1.tbl
landfilenames             windturbines.txt
LANDUSE.TBL               wrf.exe
masses.asc
seaclid@node6:~/dan1/wrf/WRF$

```

You can choose to run WRF from either directory.

### 3.4. Building WPS

- After the WRF model is built, the next step is building the WPS program (if you plan to run real cases, as opposed to idealized cases). The WRF model MUST be properly built prior to trying to build the WPS programs. You can obtain the WPS code by following the same directions for obtaining WRF. [http://www2.mmm.ucar.edu/wrf/users/download/get\\_source.html](http://www2.mmm.ucar.edu/wrf/users/download/get_source.html).

- Go into the WPS directory:

```
cd WPS
```

- Similar to the WRF model, make sure the WPS directory is clean, by issuing:

```
./clean
```

- The next step is to configure WPS, however, you first need to set some paths for the ungrib libraries:

```
setenv JASPERLIB $DIR/grib2/lib
setenv JASPERINC $DIR/grib2/include
```

- and then you can configure:

```
./configure
```

```

2. latihan
Re-attach Fullscreen Stay on top Duplicate Hide toolbar Close
seaclid@node6:~/dan1/wrf/WPS$ ./configure
Will use NETCDF in dir: /home/seaclid/Install_Library/
$JASPERLIB or $JASPERINC not found in environment. Using default values for library paths
...
Please select from among the following supported platforms.

1. Linux x86_64, gfortran (serial)
2. Linux x86_64, gfortran (serial_NO_GRIB2)
3. Linux x86_64, gfortran (dmpar)
4. Linux x86_64, gfortran (dmpar_NO_GRIB2)
5. Linux x86_64, PGI compiler (serial)
6. Linux x86_64, PGI compiler (serial_NO_GRIB2)
7. Linux x86_64, PGI compiler (dmpar)
8. Linux x86_64, PGI compiler (dmpar_NO_GRIB2)
9. Linux x86_64, PGI compiler, SGI MPT (serial)
10. Linux x86_64, PGI compiler, SGI MPT (serial_NO_GRIB2)
11. Linux x86_64, PGI compiler, SGI MPT (dmpar)
12. Linux x86_64, PGI compiler, SGI MPT (dmpar_NO_GRIB2)
13. Linux x86_64, IA64 and Opteron (serial)
14. Linux x86_64, IA64 and Opteron (serial_NO_GRIB2)
15. Linux x86_64, IA64 and Opteron (dmpar)
16. Linux x86_64, IA64 and Opteron (dmpar_NO_GRIB2)
17. Linux x86_64, Intel compiler (serial)
18. Linux x86_64, Intel compiler (serial_NO_GRIB2)
19. Linux x86_64, Intel compiler (dmpar)
20. Linux x86_64, Intel compiler (dmpar_NO_GRIB2)
21. Linux x86_64, Intel compiler, SGI MPT (serial)
22. Linux x86_64, Intel compiler, SGI MPT (serial_NO_GRIB2)
23. Linux x86_64, Intel compiler, SGI MPT (dmpar)
24. Linux x86_64, Intel compiler, SGI MPT (dmpar_NO_GRIB2)
25. Linux x86_64, Intel compiler, IBM POE (serial)
26. Linux x86_64, Intel compiler, IBM POE (serial_NO_GRIB2)
27. Linux x86_64, Intel compiler, IBM POE (dmpar)
28. Linux x86_64, Intel compiler, IBM POE (dmpar_NO_GRIB2)
29. Linux x86_64 g95 compiler (serial)
30. Linux x86_64 g95 compiler (serial_NO_GRIB2)
31. Linux x86_64 g95 compiler (dmpar)
32. Linux x86_64 g95 compiler (dmpar_NO_GRIB2)
33. Cray XE/XC CLE/Linux x86_64, Cray compiler (serial)
34. Cray XE/XC CLE/Linux x86_64, Cray compiler (serial_NO_GRIB2)
35. Cray XE/XC CLE/Linux x86_64, Cray compiler (dmpar)
36. Cray XE/XC CLE/Linux x86_64, Cray compiler (dmpar_NO_GRIB2)
37. Cray XC CLE/Linux x86_64, Intel compiler (serial)
38. Cray XC CLE/Linux x86_64, Intel compiler (serial_NO_GRIB2)
39. Cray XC CLE/Linux x86_64, Intel compiler (dmpar)
40. Cray XC CLE/Linux x86_64, Intel compiler (dmpar_NO_GRIB2)

Enter selection [1-40] : 

```

You should be given a list of various options for compiler types, whether to compile in serial or parallel, and whether to compile *ungrib* with GRIB2 capability. Unless you plan

to create extremely large domains, it is recommended to compile WPS in serial mode, regardless of whether you compiled WRF in parallel. It is also recommended that you choose a GRIB2 option (make sure you do not choose one that states "NO\_GRIB2"). You may choose a non-grib2 option, but most data are now in grib2 format, so it is best to choose this option. You can still run grib1 data when you have built with grib2.

Choose the option that lists a compiler to match what you used to compile WRF, serial, and grib2. **\*\*Note: The option number will likely be different than the number you chose to compile WRF**

the `metgrid.exe` and `geogrid.exe` programs rely on the WRF model's I/O libraries. There is a line in the `configure.wps` file that directs the WPS build system to the location of the I/O libraries from the WRF model:

```
WRF_DIR = ../WRF
```

```
2. latihan
Re-attach Fullscreen Stay on top Duplicate Hide toolbar Close
seaclid@node6:~/dan1/wrf/WPS$ ./configure
Will use NETCDF in dir: /home/seaclid/Install_Library/
$JASPERLIB or $JASPERINC not found in environment. Using default values for library paths
...
-----
Please select from among the following supported platforms.

1. Linux x86_64, gfortran (serial)
2. Linux x86_64, gfortran (serial_NO_GRIB2)
3. Linux x86_64, gfortran (dmpar)
4. Linux x86_64, gfortran (dmpar_NO_GRIB2)
5. Linux x86_64, PGI compiler (serial)
6. Linux x86_64, PGI compiler (serial_NO_GRIB2)
7. Linux x86_64, PGI compiler (dmpar)
8. Linux x86_64, PGI compiler (dmpar_NO_GRIB2)
9. Linux x86_64, PGI compiler, SGI MPT (serial)
10. Linux x86_64, PGI compiler, SGI MPT (serial_NO_GRIB2)
11. Linux x86_64, PGI compiler, SGI MPT (dmpar)
12. Linux x86_64, PGI compiler, SGI MPT (dmpar_NO_GRIB2)
13. Linux x86_64, IA64 and Opteron (serial)
14. Linux x86_64, IA64 and Opteron (serial_NO_GRIB2)
15. Linux x86_64, IA64 and Opteron (dmpar)
16. Linux x86_64, IA64 and Opteron (dmpar_NO_GRIB2)
17. Linux x86_64, Intel compiler (serial)
18. Linux x86_64, Intel compiler (serial_NO_GRIB2)
19. Linux x86_64, Intel compiler (dmpar)
20. Linux x86_64, Intel compiler (dmpar_NO_GRIB2)
21. Linux x86_64, Intel compiler, SGI MPT (serial)
22. Linux x86_64, Intel compiler, SGI MPT (serial_NO_GRIB2)
23. Linux x86_64, Intel compiler, SGI MPT (dmpar)
24. Linux x86_64, Intel compiler, SGI MPT (dmpar_NO_GRIB2)
25. Linux x86_64, Intel compiler, IBM POE (serial)
26. Linux x86_64, Intel compiler, IBM POE (serial_NO_GRIB2)
27. Linux x86_64, Intel compiler, IBM POE (dmpar)
28. Linux x86_64, Intel compiler, IBM POE (dmpar_NO_GRIB2)
29. Linux x86_64 g95 compiler (serial)
30. Linux x86_64 g95 compiler (serial_NO_GRIB2)
31. Linux x86_64 g95 compiler (dmpar)
32. Linux x86_64 g95 compiler (dmpar_NO_GRIB2)
33. Cray XE/XC CLE/Linux x86_64, Cray compiler (serial)
34. Cray XE/XC CLE/Linux x86_64, Cray compiler (serial_NO_GRIB2)
35. Cray XE/XC CLE/Linux x86_64, Cray compiler (dmpar)
36. Cray XE/XC CLE/Linux x86_64, Cray compiler (dmpar_NO_GRIB2)
37. Cray XC CLE/Linux x86_64, Intel compiler (serial)
38. Cray XC CLE/Linux x86_64, Intel compiler (serial_NO_GRIB2)
39. Cray XC CLE/Linux x86_64, Intel compiler (dmpar)
40. Cray XC CLE/Linux x86_64, Intel compiler (dmpar_NO_GRIB2)

Enter selection [1-40] : 3
```

Above is the default setting. As long as the name of the WRF model's top-level directory is "WRF" and the WPS and WRF directories are at the same level (which they should be if you have followed exactly as instructed on this page so far), then the existing default setting is correct and there is no need to change it. If it is not correct, you must modify the configure file and then save the changes before compiling.

```
2. latihan
Re-attach Fullscreen Stay on top Duplicate Hide toolbar Close

4. Linux x86_64, gfortran (dmpar_NO_GRIB2)
5. Linux x86_64, PGI compiler (serial)
6. Linux x86_64, PGI compiler (serial_NO_GRIB2)
7. Linux x86_64, PGI compiler (dmpar)
8. Linux x86_64, PGI compiler (dmpar_NO_GRIB2)
9. Linux x86_64, PGI compiler, SGI MPT (serial)
10. Linux x86_64, PGI compiler, SGI MPT (serial_NO_GRIB2)
11. Linux x86_64, PGI compiler, SGI MPT (dmpar)
12. Linux x86_64, PGI compiler, SGI MPT (dmpar_NO_GRIB2)
13. Linux x86_64, IA64 and Opteron (serial)
14. Linux x86_64, IA64 and Opteron (serial_NO_GRIB2)
15. Linux x86_64, IA64 and Opteron (dmpar)
16. Linux x86_64, IA64 and Opteron (dmpar_NO_GRIB2)
17. Linux x86_64, Intel compiler (serial)
18. Linux x86_64, Intel compiler (serial_NO_GRIB2)
19. Linux x86_64, Intel compiler (dmpar)
20. Linux x86_64, Intel compiler (dmpar_NO_GRIB2)
21. Linux x86_64, Intel compiler, SGI MPT (serial)
22. Linux x86_64, Intel compiler, SGI MPT (serial_NO_GRIB2)
23. Linux x86_64, Intel compiler, SGI MPT (dmpar)
24. Linux x86_64, Intel compiler, SGI MPT (dmpar_NO_GRIB2)
25. Linux x86_64, Intel compiler, IBM POE (serial)
26. Linux x86_64, Intel compiler, IBM POE (serial_NO_GRIB2)
27. Linux x86_64, Intel compiler, IBM POE (dmpar)
28. Linux x86_64, Intel compiler, IBM POE (dmpar_NO_GRIB2)
29. Linux x86_64 g95 compiler (serial)
30. Linux x86_64 g95 compiler (serial_NO_GRIB2)
31. Linux x86_64 g95 compiler (dmpar)
32. Linux x86_64 g95 compiler (dmpar_NO_GRIB2)
33. Cray XE/XC CLE/Linux x86_64, Cray compiler (serial)
34. Cray XE/XC CLE/Linux x86_64, Cray compiler (serial_NO_GRIB2)
35. Cray XE/XC CLE/Linux x86_64, Cray compiler (dmpar)
36. Cray XE/XC CLE/Linux x86_64, Cray compiler (dmpar_NO_GRIB2)
37. Cray XC CLE/Linux x86_64, Intel compiler (serial)
38. Cray XC CLE/Linux x86_64, Intel compiler (serial_NO_GRIB2)
39. Cray XC CLE/Linux x86_64, Intel compiler (dmpar)
40. Cray XC CLE/Linux x86_64, Intel compiler (dmpar_NO_GRIB2)

Enter selection [1-40] : 3
-----
Configuration successful. To build the WPS, type: compile
-----

Testing for NetCDF, C and Fortran compiler

This installation NetCDF is 64-bit
C compiler is 64-bit
Fortran compiler is 64-bit

seaclid@node6:~/dan1/wrf/WPS$
```

- You can now compile WPS:

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



```
2. latihan
Re-attach Fullscreen Stay on top Duplicate Hide toolbar Close

4. Linux x86_64, gfortran (dmpar_NO_GRIB2)
5. Linux x86_64, PGI compiler (serial)
6. Linux x86_64, PGI compiler (serial_NO_GRIB2)
7. Linux x86_64, PGI compiler (dmpar)
8. Linux x86_64, PGI compiler (dmpar_NO_GRIB2)
9. Linux x86_64, PGI compiler, SGI MPT (serial)
10. Linux x86_64, PGI compiler, SGI MPT (serial_NO_GRIB2)
11. Linux x86_64, PGI compiler, SGI MPT (dmpar)
12. Linux x86_64, PGI compiler, SGI MPT (dmpar_NO_GRIB2)
13. Linux x86_64, IA64 and Opteron (serial)
14. Linux x86_64, IA64 and Opteron (serial_NO_GRIB2)
15. Linux x86_64, IA64 and Opteron (dmpar)
16. Linux x86_64, IA64 and Opteron (dmpar_NO_GRIB2)
17. Linux x86_64, Intel compiler (serial)
18. Linux x86_64, Intel compiler (serial_NO_GRIB2)
19. Linux x86_64, Intel compiler (dmpar)
20. Linux x86_64, Intel compiler (dmpar_NO_GRIB2)
21. Linux x86_64, Intel compiler, SGI MPT (serial)
22. Linux x86_64, Intel compiler, SGI MPT (serial_NO_GRIB2)
23. Linux x86_64, Intel compiler, SGI MPT (dmpar)
24. Linux x86_64, Intel compiler, SGI MPT (dmpar_NO_GRIB2)
25. Linux x86_64, Intel compiler, IBM POE (serial)
26. Linux x86_64, Intel compiler, IBM POE (serial_NO_GRIB2)
27. Linux x86_64, Intel compiler, IBM POE (dmpar)
28. Linux x86_64, Intel compiler, IBM POE (dmpar_NO_GRIB2)
29. Linux x86_64 g95 compiler (serial)
30. Linux x86_64 g95 compiler (serial_NO_GRIB2)
31. Linux x86_64 g95 compiler (dmpar)
32. Linux x86_64 g95 compiler (dmpar_NO_GRIB2)
33. Cray XE/XC CLE/Linux x86_64, Cray compiler (serial)
34. Cray XE/XC CLE/Linux x86_64, Cray compiler (serial_NO_GRIB2)
35. Cray XE/XC CLE/Linux x86_64, Cray compiler (dmpar)
36. Cray XE/XC CLE/Linux x86_64, Cray compiler (dmpar_NO_GRIB2)
37. Cray XC CLE/Linux x86_64, Intel compiler (serial)
38. Cray XC CLE/Linux x86_64, Intel compiler (serial_NO_GRIB2)
39. Cray XC CLE/Linux x86_64, Intel compiler (dmpar)
40. Cray XC CLE/Linux x86_64, Intel compiler (dmpar_NO_GRIB2)

Enter selection [1-40] : 3
-----
Configuration successful. To build the WPS, type: compile
-----

Testing for NetCDF, C and Fortran compiler

This installation NetCDF is 64-bit
C compiler is 64-bit
Fortran compiler is 64-bit

seaclid@node6:~/dan1/wrf/WPS$ ./compile >& log
```

Compilation should only take a few minutes.

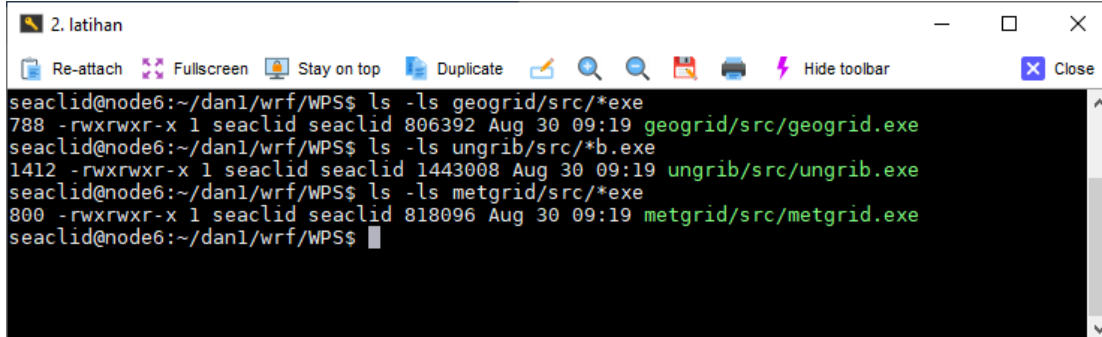
- If the compilation is successful, there should be 3 executables in the WPS top-level directory, that are linked to their corresponding src/ directories:

```
geogrid.exe -> geogrid/src/geogrid.exe
ungrib.exe -> ungrib/src/ungrib.exe
metgrid.exe -> metgrid/src/metgrid.exe
```

Verify that they are not zero-sized (inside the \*/src/ directory). To see file size, you can type (for e.g.):



```
ls -ls geogrid/src/geogrid.exe
```



```
2. latihan
Re-attach Fullscreen Stay on top Duplicate
seaclid@node6:~/dan1/wrf/WPS$ ls -ls geogrid/src/*exe
788 -rwxrwxr-x 1 seaclid seaclid 806392 Aug 30 09:19 geogrid/src/geogrid.exe
seaclid@node6:~/dan1/wrf/WPS$ ls -ls ungrib/src/*b.exe
1412 -rwxrwxr-x 1 seaclid seaclid 1443008 Aug 30 09:19 ungrib/src/ungrib.exe
seaclid@node6:~/dan1/wrf/WPS$ ls -ls metgrid/src/*exe
800 -rwxrwxr-x 1 seaclid seaclid 818096 Aug 30 09:19 metgrid/src/metgrid.exe
seaclid@node6:~/dan1/wrf/WPS$
```

## Static Geography Data

- The WRF modeling system is able to create idealized simulations, though most users are interested in the real-data cases. To initiate a real-data case, the domain's physical location on the globe and the static information for that location must be created. This requires a data set that includes such fields as topography and land use categories. These data are available from the WRF download page

[http://www2.mmm.ucar.edu/wrf/users/download/get\\_sources\\_wps\\_geog.html](http://www2.mmm.ucar.edu/wrf/users/download/get_sources_wps_geog.html)

- Download the file and place it in the `Build_WRF` directory. Keep in mind that if you are downloading the complete dataset, the file is very large. If you are sharing space on a cluster, you may want to consider placing this in a central location so that everyone can use it, and it's not necessary to download for each person. Uncompressing and un-tar the file:

```
gunzip geog.tar.gz
tar -xf geog.tar
```

- When you *untar* the file, it will be called "geog." Rename the file to "WPS\_GEOG."

```
mv geog WPS_GEOG
```

- The directory information is given to the geogrid program in the namelist.wps file in the `&geogrid` section:

```
geog_data_path = 'path_to_directory/Build_WRF/WPS_GEOG'
```

- The data expands to approximately 10 GB. This data allows a user to run the `geogrid.exe` program.

## Real-time Data

- For real-data cases, the WRF model requires up-to-date meteorological information for both an initial condition and also for lateral boundary conditions. This meteorological data is traditionally a *Grib* file that is provided by a previously run external model or analysis. For a semi-operational set-up, the meteorological data is usually sourced from a global model, which permits locating the WRF model's domains anywhere on the globe.
- The National Centers for Environmental Prediction (NCEP) run the Global Forecast System (GFS) model four times daily (initializations valid for 0000, 0600, 1200, and 1800 UTC). This is a global, isobaric, 0.5-degree latitude/longitude, forecast data set that is freely available, and is usually accessible +4h after the initialization time period.
- A single data file needs to be acquired for each requested time period. For example, if we would like hours 0, 6, and 12 of a forecasts that is initialized 2019 July 12 at 0000 UTC, we need the following times:

2019071200 – 0 h

2019071206 – 6 h

2019071212 – 12 h

These translate to the following file names to access:

`gfs.2019071200/gfs.t00z.pgrb2.0p50.f000`

`gfs.2019071200/gfs.t00z.pgrb2.0p50.f006`

`gfs.2019071200/gfs.t00z.pgrb2.0p50.f012`

Note that the initialization data and time (`gfs.2019071200`) remains the same, and that the forecast cycle remains the same (`t00z`). What is incremented is the forecast hour (`f00`, `f06`, `f12`).

### 3.5. Run WPS and WRF

#### Running WPS

- You are now ready to begin running WPS and WRF. Start by going to the WPS directory:

```
cd WPS
```

- Make any changes to the `namelist.wps` file, to reflect information for your particular run

- Before running geogrid, make sure that you have your `geog_data_path` set to the location where you put your geography static data. Once that is set, you can run geogrid.

```
./geogrid.exe >& log.geogrid
```

- If you successfully created a `geo_em*` file for each domain, then you are ready to prepare to run ungrib. Start by linking in the input GFS data:

```
./link_grib.csh path_where_you_placed_GFS_files
```

Then link to the correct Vtable (GFS, for this case):

```
ln -sf ungrib/Variable_Tables/Vtable.GFS Vtable
```

Then run the ungrib executable:

```
./ungrib.exe
```

You should now have files with the prefix "FILE" (or if you named them something else, they should have that prefix)

You are now ready to run metgrid:

```
./metgrid.exe >& log.metgrid
```

You should now have files with the prefix `met_em*` for each of the time periods for which you are running.

## Running WRF

- You are now ready to run WRF. Move into the `WRF` directory, and then into either the `run/` directory, or the `test/em_real/`

directory:

```
cd ../WRF/run
```

or

```
cd ../WRF/test/em_real
```

Before running the "real" program, you need to make all necessary changes to reflect your particular case to the `namelist.input` file. Once that is complete, you need to copy or link your `met_em*` files into the working directory:

```
ln -sf ../../../../WPS/met_em* . (from  
the test/em_real directory), or
```

```
ln -sf ../../WPS/met_em* . (from the run/ directory).
```

or, if you would rather copy the files in, instead of linking them, you can use the `cp` command, instead of the `ln -sf` command.

- You can now run the "real" program. The command for running this may vary depending on your system and the number of processors you have available, but it should be something similar to:

```
mpirun -np 1 ./real.exe
```

Check the end of your "rsl" files to make sure the run was successful:

```
tail rsl.error.0000
```

If you see a "SUCCESS" in there, and you see a `wrfbdy_d01` file, and `wrfinput_d0*` files for each of your domains, then the run was successful.

- To run WRF, type something similar to:

```
mpirun -np 8 ./wrf.exe
```

Again, check your "rsl" file for "SUCCESS", and make sure you have all the `wrfout*` files you anticipated having. If so, the run was successful, and you are ready to do analysis for your project.

## Post-Processing of WRF model output

Once the model run is over, we will move towards doing the post-processing and generating the output. The program ARWpost will allow reading the wrf output file (NetCDF format) and convert it to Grads readable format (Binary format). ARWpost is a Fortran program that reads WRF-ARW input and output files, then generates GrADS output files. Once the output files have been generated, GrADS can be used to produce horizontal or vertical crosssection plots of scalar fields (contours) or vector fields (barbs or arrows), vertical profiles and soundings.

For the post processing to start first let's move to directory where we will do the post processing. Issue command

```
cd /home/user/ARWpost
```

to move to the directory.

Before starting the conversion process, we have to set the parameters we want to do post processing for. We can do that by modifying the `namelist.ARWpost` residing inside the ARWpost directory. For precipitation we need to activate the `RAINC` and `RAINNC` options and for mean sea level pressure we need to activate the `slvl` parameter. Other parameters those can be utilized can be checked from the file and the parameters have the explanation against them too. So, we can activate as much parameters we want to see output for. The other important job to do is to define the absolute path to the wrf output file, just generated which would be

```
wrfout_d01_YYYY-mm-dd:00:00
```

To edit the `control_file` please issue command

```
vi namelist.ARWpost
```

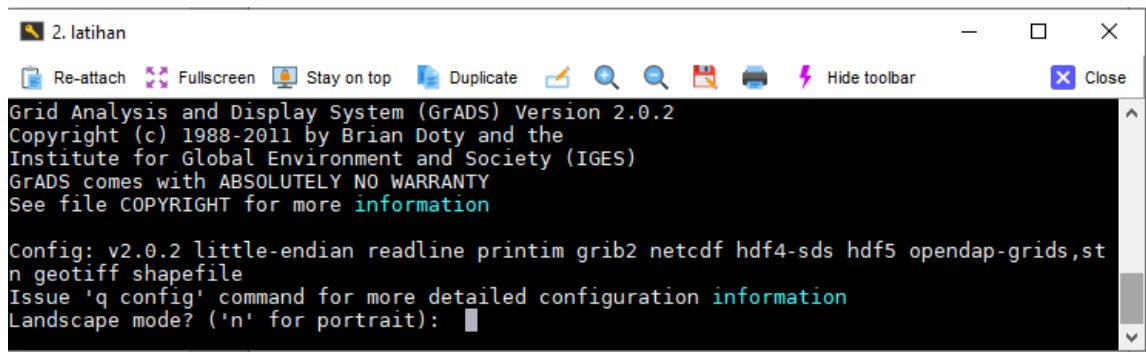
Once the editing is over we can now give the command on the command line:

```
./ARWpost.exe
```

Once the command is executed the process should generate two files `test.ctl` and another `test.dat` file, if completed successfully. The `ctl` file is a description file with detailed information about the data the `filename.dat` contains and the `filename.dat` file contains the actual binary data.

### **PLOT with GrADS**

With the data in hand, we can plot the weather advisories for different components through GrADS and we can save the output to various formats according to requirement. Staying inside the ARWpost directory, user has to issue command `grads` at the command prompt. It should ask for the user input for landscape or portrait view of the window. Default will be on landscape mode if you won't choose any option. Choosing option 'no' will open in portrait mode.



```
2. latihan
Grid Analysis and Display System (GrADS) Version 2.0.2
Copyright (c) 1988-2011 by Brian Doty and the
Institute for Global Environment and Society (IGES)
GrADS comes with ABSOLUTELY NO WARRANTY
See file COPYRIGHT for more information

Config: v2.0.2 little-endian readline printim grib2 netcdf hdf4-sds hdf5 opendap-grids, st
n geotiff shapefile
Issue 'q config' command for more detailed configuration information
Landscape mode? ('n' for portrait):
```

Upon choosing the option it will open grads and the prompt will change to `ga>` mode, where it will only accept GrADS known commands. Basic commands will be opening a description file. Viewing file details, setting parameters and plotting.

To open a file issue command at `ga>` prompt

```
open test.ct1
```

where `test.ct1` is the control file generated using ARWpost

To see all parameters in the file

```
q file
```

```
d parameter to plot parameters
```

```
set properties value for setting properties for plot parameters.
```

## Exercise

1. Mention the reasons why the process is necessary before the WRF installation process?
2. In the WRF installation process, what libraries need to be defined first?
3. Can you specify what files need to be edited in addition to input data before running WRF?

## CHAPTER IV. SIMULATION PROCEDURES

**Objective** : Participants are able to apply basic skills in developing WRF modeling and its application.

In several design projects, current architectural and engineering practice requires careful consideration of local meteorology as a key factor. Parameters such as wind speed, temperature, humidity, rainfall, and incoming radiation can all affect the design and ongoing performance of the structure. The Global Forecast System (GFS) is a model of weather forecasts provided by the National Environmental Prediction Centers (NCEP). Through this dataset, hundreds of atmospheric and land-soil variables are available, from temperatures, winds, and precipitation to soil moisture and ozone concentration in the atmosphere. However, these data are usually collected from meteorological stations that may be tens of hundreds of kilometers away from a study site, or even in entirely different terrain (urban versus rural, mountain versus valley), which may not be representative.

In the absence of an adequate observational station, there are a number of techniques available for deriving the meteorology of a given site. Such methods also consist of some form of interpolation or more complex physical aerodynamic process/modeling to downscale information to a finer resolution. The interpolation method can be as simple or more complex as spatial linear interpolation of adjacent observation stations, such as constructing linear regression models based on predictors that account for elevation or proximity to a coast. (Daly et al. 2008).

Mesoscale weather models, such as the Weather Research Forecast (WRF) model, consider the preservation of mass, momentum, energy and humidity at the far end of sophistication, combined with representations of atmospheric radiation and cloud formation. These mesoscale models represent the meteorology of grids approaching the microscale (i.e., 1 km horizontally), usually used by meteorologists and climatologists. While these mesoscale models can be used to refine historical climate data at a very high spatial resolution, the sophistication and strength of the mesoscale model is a barrier to entry for first-time users; without thoroughly understanding the physics schemes and dynamic mechanisms used for different weather conditions, it can be difficult to implement the model.

The WRF model is a numerical weather prediction system of the next generation mesoscale designed to serve both operational forecasting and research needs in the atmosphere. The

model is sufficient for a wide variety of applications ranging from meters to thousands of kilometers across scales. The WRF model is created and maintained predominantly by U.S. government agencies, universities, national laboratories, and international communities as part of a collaborative effort (Skamarock et al. 2008).

#### **4.1. Domain Configuration**

With the development of the model domain, the modeling process starts. The mesoscale modeling is three-dimensional; the method takes the atmosphere into 35 vertical layers, and horizontally by grid cells in kilometers spanning an entire domain, from the ground surface to the top of the troposphere, about 100 hPa.

The WRF model adopts a nested approach where a large domain of coarse resolution feeds into a domain of small but fine resolution. The parent or first domain, for instance, spans about 1800 x 1800 km with cells of 36 x 36 km. A medium-resolution child domain, which is 730 x 730 km, is nestled in this domain. An even finer-resolution child domain, which is 280 x 280 km, is nested in this domain. The final grid resolution is approximately 4 km for the smallest domain.

#### **4.2. Input Data**

The WRF model must be combined with a larger three-dimensional reanalysis data set output, one with greater area coverage. The larger model is interpolated into the smaller regional model and acts as the initial atmospheric state (the initial condition). The larger model also provides the air masses that join the domain along the lateral borders as the simulation evolves (the boundary conditions). Finally, the global model often limits the coarsest domain in such a way that the simulation of the model does not drift too far from reality; this is called nudging. In order to provide improved precision and solution stability, the WRF model can also assimilate available observational data from the upper-air and surface-station.

#### **4.3. Model Physics**

Based on a series of sensitivity simulations, the physical modules used in the model were selected to provide a solution that adequately represents the regions studied:

Cumulus parameterization: The Kain-Fritsch scheme (Kain and Fritsch 1993) was applied to coarse WRF domains (36 and 12 km) to parameterize or reflect cloud processes that could not be clearly represented on those scales. For the fine 4 km domains, however, this parameterization was turned off as clouds are partly clearly resolved at this scale.



Planetary boundary layer scheme: Planetary boundary layer (PBL) physics models predict low-level wind, temperature, turbulence, cloud cover, and radiation. The Yonsei University (YSU) (Hong et al. 2006) scheme has been chosen.

Explicit moisture method: The explicit moisture scheme is used from the inside of a grid cell for the fine grids to address cloud formation and precipitation (4 km). Here, we used the single-moment WSM6 scheme (Hong and Lim 2006).

Radiation schemes: The rapid radiative transfer model (RRTM) (Mlawer et al. 1997) was used for longwave radiation. The Dudhia method (Dudhia 1989) was used for shortwave radiation.

Land surface scheme: The generally used unified Noah land surface model (Chen and Dudhia 2001), a five-layer thermal diffusion scheme, was used to simulate the temperature of the surface skin.

It should be noted that for a given region, it is likely that local climate conditions may be better reflected by a different set. The solar radiation schemes suggested here, for example, do not involve aerosol and ozone regional impacts. There are far more complex schemes available if solar radiation is of major importance.

#### **4.4. Model Simulation**

The three-day periods are determined as an example run. The spin-up added is a 12-hour part on the front of this simulation that enables the model to achieve a balanced state with the boundary conditions. A realistic flow field evolves beyond this spin-up period, from which a functional portion of the simulation is constructed.

#### **4.5. Postprocessing**

A large collection of files consisting of hourly three-dimensional fields are the output of the WRF simulations. As the ultimate objective of the technique is to create a time series on the surface of the required meteorological quantities, these three-dimensional files need to be mined.

### **Exercise**

1. What needs to be prepared before running the WRF model?
2. Why do local conditions need to be considered in the process of running WRF?

## **CHAPTER V. SUMMARY**

### **5.1. Summary**

This chapter builds on the information presented in the preceding chapters to provide guidance for main point to be learn on previous chapter. The WRF model is a numerical weather prediction system of the next generation mesoscale developed for both atmospheric analysis and operational forecasting applications.

The treatment of unresolvable physical processes, otherwise known as parameterization, is one of the most difficult aspects of modeling the climate system. Over a very large range of time and space scales, atmospheric processes work. Parameterization is how indirectly we include the results of physical processes although we do not directly include the processes themselves. Instead of simulation, parameterization can be seen as emulation (modeling the results of a process).

We will shift toward doing the post-processing and generating the output once the model run is done. The ARWpost platform allows the WRF output file to be read and translated to Grads readable format. ARWpost is a Fortran program that reads input and output files for WRF-ARW, then produces output files for GrADS. GrADS can be used for producing horizontal or vertical cross-section plots of scalar fields (contours) or vector fields (barbs or arrows), vertical profiles and soundings once the output files have been produced.

### **5.2. Recommendation**

Current architecture and engineering practice needs careful consideration of local meteorology as a key element in many design projects. The structure's design and ongoing efficiency can all be influenced by parameters such as wind speed, temperature, humidity, rainfall and incoming radiation. There are a variety of techniques available for deriving the meteorology of a given site in the absence of an appropriate observational station. Such techniques often consist of some form of interpolation or more complex aerodynamic physical process/modeling to downscale data to a finer resolution.

## REFERENCES

- Chen, F., and J. Dudhia. 2001. Coupling an advanced land-surface/hydrology model with the Penn State/NCARMM5 modeling system. Part I: Model description and implementation. *Monthly Weather Review* 129:569–85.
- Daly, C., M. Halbleib, J.I. Smith, W.P. Gibson, M.K. Doggett, G.H. Taylor, J. Curtis, and P.P. Pasteris. 2008. Physiographically sensitive mapping of climatological temperature and precipitation across the conterminous United States. *International Journal of Climatology* 28:2031–64.
- Dudhia, J. 1989. Numerical study of convection observed during the winter monsoon experiment using a meso-scale two-dimensional model. *Journal of the Atmospheric Sciences* 46:3077–3107
- Hong, S.Y., and J.O.J. Lim. 2006. The WRF single-moment6-class microphysics scheme (WSM6). *Journal of the Korean Meteorological Society* 42:129–51.
- Hong, S.Y., Y. Noh, and J. Dudhia. 2006. A new vertical diffusion package with explicit treatment of entrainment processes. *Monthly Weather Review* 134:2318–41.
- Kain, J.S., and J.M. Fritsch. 1993. Convective parameterization for mesoscale models: The Kain-Fritsch scheme..The representation of cumulus convection in numerical models. Boston: American Meteorological Society, pp. 165–170
- Skamarock, W.C., J.B. Klemp, J. Dudhia, D.O. Gill, D.M. Barker, M.G. Duda, X.Y. Huang, W. Wang, and J.G. Powers. 2008. A description of the advanced research WRF Version 3. NCAR/TN–475+STR, NCAR Technical Note. Boulder, CO: NCAR.