User's Guide for the NMM Core of the Weather Research and Forecast (WRF) Modeling System Version 2.1

Chapter 2: Software Installation

Table of Contents

Introduction

The WRF modeling system software installation is fairly straightforward on the platforms. The package is mostly self-contained, meaning that WRF requires no external libraries (such as for FFTs or various linear algebra solvers). The one required external package is the netCDF library, which is one of the supported I/O API packages. The netCDF libraries or source code are available from the Unidata homepage at http://www.unidata.ucar.edu (select DOWNLOADS, registration required).

The WRF-NMM core has been successfully ported to a number of Unix-based machines. WRF developers do not have access to all of them and must rely on outside users and vendors to supply the required configuration information for the compiler and loader options. Below is a list of the supported combinations of hardware and software for WRF-NMM.

Vendor	Hardware	O.S.	Compiler
IBM	SP Power-*x*	AIX	vendor
SGI	MIPS	IRIX	vendor
HP/COMPAQ/DEC	Alpha	Tru64	vendor
Various	IA-32	LINUX	PGI

Various	Opteron	LINUX	PGI

The WRF-NMM code runs on single processor machines (if number of processors for "mpirun" is set to 1), shared-memory machines (that use the OpenMP API), distributed memory machines (with the appropriate MPI libraries), and on distributed clusters (utilizing both OpenMP and MPI). Porting to systems that uses the Intel compiler is currently under development.

The WRF-NMM SI code also runs on the systems listed above.

Required Compilers and Scripting Languages

The WRF model is written in FORTRAN (what many refer to as FORTRAN 90). The software layers, RSL and RSL-LITE, -which sit between WRF and the MPI interface- are written in C. Ancillary programs that perform file parsing and file construction, both of which are required for default building of the WRF modeling code, are written in C. Additionally, the WRF build mechanism uses several scripting languages: including perl (to handle various tasks such as the code browser designed by Brian Fiedler), C-shell and Bourne shell. The traditional UNIX text/file processing utilities are used: ***make, M4, sed***, and ***awk***. See Chapter 6: WRF Software (Required Software) for a more detailed listing of the necessary pieces for the WRF build.

The WRF-NMM SI is mostly written in FORTRAN 77 and FORTRAN 90 with a few C routines. Perl scripts are used to run the programs, and Perl/Tk is used for GUI.

UNIX make is used in building all executables.

Required/Optional Libraries to Download

The only library that is *almost always* required is the netCDF package from Unidata (login > Downloads > NetCDF). The WRF post-processing packages assume that the data from the WRF model uses the netCDF libraries. To execute netcdf commands, such as **ncdump** and **ncgen,** /path-to-netcdf/netcdf/bin may also need to be added to the user's path.

Hint: When compiling WRF codes on a Linux system using the PGI compiler, make sure the netCDF library is also installed using the PGI compiler.

A version of MPI is needed when running distributed memory WRF jobs. A version of mpich can be picked up at (insert web link), but the user may want their system administrator to install the code. A working installation of MPI is required prior to a build of WRF using distributed memory. To determine whether MPI is available on your computer system, try:

which mpif90
which mpicc

which mpirun

If all of these executables are defined, MPI is probably already available. The MPI lib, include and bin need to be included in the user's path.

Post-Processing Utilities

The currently supported graphical utilities for the WRF-NMM require first running the raw output files through the WRF Post-processing package developed by NCEP. This package interpolates the output from the model's native grid to standard output levels and grids, as well as computing some diagnostic fields.

Some characteristics of the WRF Post-Processor are:

- outputs the results in NWS and WMO standard GRIB1 format
- interpolates the forecasts from the model's native vertical coordinate to NWS standard output levels (pressure, height, etc.) and computes MSLP.
- computes diagnostic output quantities (e.g. CAPE, helicity, radar reflectivity, etc.).
- destaggers forecasts.

- GrADS
  - interpolates to regular lat/lon grid
  - simple to generate publication quality

- GEMPAK
  - distributed and supported to user community by UNIDATA
  - interpolation to various surfaces, trajectories, hundreds of diagnostic calculations
  - table driven
  - interpolates to regular lat/lon grid
  - simple to generate publication quality

UNIX Environment Settings

There are only a few WRF-related environmental settings. Most of these settings are not required, but the user may want to try some of these settings if difficulties are encountered during the build process. The one required environmental setting when building the NMM core is WRF_NMM_CORE (see below). In C-shell syntax:

- **setenv NETCDF** /path-to-netcdf (explicitly define the path to netcdf library and include directory)
- **setenv WRF_NMM_CORE 1** (explicitly defines which model core to build)
- **unset limits** (especially if you are on a small system)

- **setenv MP_STACK_SIZE 64000000** (OpenMP blows through the stack size, set it large)
- **setenv MPICH_F90 f90** (or whatever your FORTRAN compiler may be called. WRF needs the bin, lib, and include directories)
- **setenv OMP_NUM_THREADS *n*** (where *n* is the number of processors to use. In systems with OpenMP installed, this is how the number of threads is specified.)

Building the WRF System for the NMM Core

The WRF code has a fairly complicated build mechanism. The package tries to determine the architecture on which the code is being built, and then presents the user with options to allow the user to select the preferred build method. For example, on a Linux machine, it determines whether the machine is 32 or 64 bit, and then prompts the user for the desired usage of processors (such as serial, shared memory, or distributed memory).

- Get the WRF zipped tar file
  - WRFV2.1.1 from ***http://www.dtcenter.org/wrf-nmm/users***
  - always get the latest version if you are not trying to continue a pre-existing project
- unzip and untar the file
  - ***tar -zxvf wrfv2.1.1.tar.gz***
  - again, always obtain the latest version of the code, 2.1.1 is just used as an example
- ***cd WRFV2***
- ***./configure***
  Chose one of the options (best to stick with an option recommended for NMM)
- ***setenv WRF_NMM_CORE 1***
- ***./compile real_nmm***
- ***ls -ls main/*.exe***
  - ***real_nmm.exe***, and ***wrf.exe*** should appear in this directory listing

Building the WRF-NMM SI Code

The simplest build for WRF-NMM SI code is to use all default directories. Several configure files are provided in src/include/ directory for various computers. A Perl script, ***install_wrfsi.pl*** in the top directory is used to install the software.

- Get the latest WRF-NMM SI zipped tar file
  - wrfsi_nmm_v2.1.tar.gz from http://wrfsi.noaa.gov/release/
- unzip and untar the file
  - ***tar -zxvf wrfsi_v2.1.tar.gz***
- ***cd wrfsi***
- set the following environment variable to define where the netCDF library and include directories are

    - ***setenv NETCDF /path-to-netcdf***
- issue the following command to install - when prompted to answer whether the GUI should be installed:
    - ***perl install_wrfsi.pl***
    - output from running the install script can be found in ***make_install.log***
- ***ls -l bin/*** (if environment variable INSTALLROOT is not set) or ***ls -l $INSTALLROOT/bin*** (if environment variable INSTALLROOT is set)

More details can be found in Chapter 3.