Script for WRF on Raspberry Pi demonstration

January 2026 – Version 1.0

Recipe for installing and running the latest version of WRF on a Raspberry Pi. This script has been tested only on a Raspberry Pi 5 with 16 GB memory. These programs are compiled and run within a miniforge environment. An OMP version of WRF wrf.exe, real.exe and an MPI version of WPS programs geogrid.exe and metgrid.exe are created. The ungrib.exe program is serial only.

This is an alternative to this project: https://github.com/NCAR/pi-wrf

*** ---------- preliminaries [done once] ---------- ***

```
# install tcsh, needed for WRF configure script
sudo apt install tcsh -y

# if ncview is desired
sudo apt install ncview -y

# download miniforge for ARM64

wget https://github.com/conda-forge/miniforge/releases/latest/download/Miniforge3-Linux-aarch64.sh -O ~/miniforge.sh [ all one line]

# run miniforge installer
bash ~/miniforge.sh -b -p $HOME/miniforge3

# initialize shell
eval "$($HOME/miniforge3/bin/conda shell.bash hook)"
conda init

# source .bashrc
source ~/.bashrc

# create new environment and activate it
conda create -n wrf_env python=3.11 -y
conda activate wrf_env

# Compilers for ARM (Pi 5)
conda install -c conda-forge gfortran_linux-aarch64 gcc_linux-aarch64 gxx_linux-aarch64 -y

# Libraries needed for WRF and wrf-python
conda install -c conda-forge netcdf-fortran netcdf4 hdf5 m4 -y
conda install -c conda-forge numpy scipy xarray netcdf4 cython cartopy matplotlib
conda install -c conda-forge metpy nco
```

```
# we need a really old jasper for WPS
conda install conda-forge::jasper==1.900.31
conda install -c conda-forge libpng zlib

# install MPI (for metgrid, geogrid)
conda install conda-forge::mpich

# install jupyter lab
conda install conda-forge::jupyterlab

# install wrf-python (adds pandas, xarray)
conda install conda-forge::wrf-python

# the gfortran compiler is called “aarch64-conda-linux-gnu-gfortran”
# create a symbolic link to “gfortran” for convenience
# enter the text shown in grey into a file called “gfortran”. Text is one line.

sudo nano /usr/local/bin/gfortran
#!/bin/bash
exec /home/$USER/miniforge3/envs/wrf_env/bin/aarch64-conda-linux-gnu-
gfortran "$@"

sudo chmod +x /usr/local/bin/gfortran

* ---------- setup environment for compilation ----------*
# root for conda environment
export CONDA_ROOT=$CONDA_PREFIX

# NetCDF
export NETCDF=$CONDA_ROOT
export NETCDFINC=$CONDA_ROOT/include
export NETCDF_LIB=$CONDA_ROOT/lib
export HDF5=$CONDA_ROOT
export NETCDF_FORTRANINC=$CONDA_ROOT/include
export NETCDF_FORTRAN_LIB=$CONDA_ROOT/lib

# compiler flags for configure
export LDFLAGS="-L$CONDA_ROOT/lib"
export CPPFLAGS="-I$CONDA_ROOT/include"

# compression libraries for WPS
export ZLIB_ROOT=$CONDA_ROOT
export JASPERLIB=$CONDA_ROOT/lib
export JASPERINC=$CONDA_ROOT/include
export PNG_LIB=$CONDA_ROOT/lib
export PNG_INC=$CONDA_ROOT/include
```

```
# modify PATH
export PATH=$CONDA_ROOT/bin:$PATH

* ---------- compile WRF ----------*
• start in ~/

# get latest version of WRF
git clone https://github.com/wrf-model/WRF.git
        → creates directory ~/WRF
cd WRF/

• run configure. This creates configure.wrf

./configure
        Select option 2
./compile em_real
→ real.exe and wrf.exe should be created in main/ and run/ directories

* ---------- compile WPS ----------*
• start in ~/

# get latest version of WPS
git clone https://github.com/wrf-model/WPS.git
        → creates directory ~/WPS
cd WPS/

• in WPS/arch, edit configure.defaults, adding "aarch64" to this line:

#ARCH    Linux aarch64 i486 i586 i686, gfortran   # serial serial_NO_GRIB2 dmpar
dmpar_NO_GRIB2

• in WPS, run configure. Select option 1 (serial) or 3 (MPI). This creates the file
configure.wps

./configure

• edit configure.wps. Add the libraries indicated in red below

WRF_LIB       =      -L$(WRF_DIR)/external/io_grib1 -lio_grib1 \
                  -L$(WRF_DIR)/external/io_grib_share -lio_grib_share \
                  -L$(WRF_DIR)/external/io_int -lwrfio_int \
                  -L$(WRF_DIR)/external/io_netcdf -lwrfio_nf \
                  -L$(NETCDF)/lib  -lnetcdff -lnetcdf -lgomp

./compile
→ geogrid.exe, metgrid.exe, and ungrib.exe should be created
```

*** ---------- preparing to run WRF/WPS ---------- ***

• From
https://www2.mmm.ucar.edu/wrf/users/download/get_sources_wps_geog.html
download at least the lowest resolution of each mandatory field file:
https://www2.mmm.ucar.edu/wrf/src/wps_files/geog_low_res_mandatory.tar.gz
→ untar this in your home directory and call the directory WPS_GEOG_LOW_RES/

***---------- running WRF/WPS ---------- ***

• This example creates a 90 km domain covering much of North America, and initializes WRF using low-resolution NNRP reanalyses for a simulation of the March 1993 "Storm of the Century"

• Make sure you are in the WRF conda environment
conda activate wrf_env

• Create a directory for your WRF simulation and move into it. Example:

mkdir -p SOC
cd SOC

• From
https://www.atmos.albany.edu/facstaff/rfovell/NWP/WRF_on_Mac/index.html
obtain these files: namelist.wps.PRISTINE, namelist.input.PRISTINE,
make_all_links.sh, NNRP_DATA.tar.gz, and plot_WRF_output.ipynb

• untar the NNRP data. This creates a folder called NNRP_199303/
tar -zxvf NNRP_DATA.tar.gz

• setup the space using make_all_links.sh, which creates copies or links to executables and support files. The make_all_links.sh file expects your WRF and WPS directories to be ~/WRF and ~/WPS, respectively. It also copies over some Vtables and other support files

sh make_all_links.sh

• copy the namelists and customize as needed
cp namelist.wps.PRISTINE namelist.wps
cp namelist.input.PRISTINE namelist.input

• make the domain. Edit namelist.wps if needed. If using low resolution topographic information, make sure geog_data_res='lowres' and not 'default'. The run geogrid

mpirun -np 4 ./geogrid.exe

• link to the NNRP data

csh link_grib.csh NNRP_199303/* .

• copy the Vtable for NNRP
cp Vtable.NNRP Vtable

• run ungrib. Never run ungrib using more than one processor
→ Should take only a few seconds for this very low resolution reanalysis

./ungrib.exe

• run metgrid. You can run metgrid on multiple processors

mpirun -np 4 ./metgrid.exe

• Edit namelist.input as necessary. Run real.exe. Real and WRF have been compiled as OpenMP so it should use 4 threads

./real.exe

• Run WRF.

./wrf.exe

*** ---------- post-processing ----------***
• The notebook plot_WRF_output.ipynb uses wrf-python to plot the snow footprint from the SOC simulation