**Hawaii Weather Research Forecast Model Data Processing Instructions**

**Alexander Keyel,** [**akeyel@albany.edu**](mailto:akeyel@albany.edu)

## Getting Started

This workflow requires R. For working with R, I highly recommend using RStudio (<https://posit.co/> now called posit, apparently).

This guide will cover 2D and 3D data products. The 2D data products are available from the USGS via the internet. The 3D data products are on physical hard-drives at the University of Hawaii Manoa. It will require a person to physically access the hard drives to extract any desired data.

All scripts (and this User’s Manual!) are available on GitHub: [www.github.com/akeyel/HI\_WRF](http://www.github.com/akeyel/HI_WRF).

In file names, ok indicates Oahu/Kauai and hm indicates Hawaii Big Island and Maui. Unfortunately the data conventions for the download differ between the two sets of islands, but once they are converted to daily files, the four islands should all use the same processing steps.

## Processing Notes

* January 1 1996 was missing from the present day runs.
  + For rainfall, for Maui and Hawaii this day was interpolated by assigning the rainfall that fell on Jan 1 and Jan 2 over the two day period evenly to the intervening hours.
  + For Oahu and Kauai, it was assumed no rainfall fell during that 24 hour period (due to the missing i\_rain variable). This could be adjusted on the daily aggregates if so desired.
* Data were processed on Hawaii time rather than GMT time.
  + Due to the GMT offset of 10 hours, the last year was missing 10 time steps.
  + These timesteps were filled in using the 364th day of the 2009 run to capture recent synoptic conditions.
* Day 365 for the year 2007 for Maui RCP 4.5 had negative rainfall and corrupted I\_RAIN values. This day’s values were replaced with those from day 364.
* The I\_RAIN variable was missing for Oahu and Kauai. We used the difference in precipitation between each hours. This will underestimate rainfall if more than 100 mm falls in a 1 hour period. (based on Maui data, that heavy of rainfall was a relatively rare event).
* For UDROFF for Kauai, negative values were observed for day 66 in 1994, day 1 in 1996, and days 175 and 176 for 1998. For 1994 and 1998, it looks like the counter rolled over back to 0, so the change was just marked as 0 instead of negative, and differences from the new baseline were used going forward. I did not carefully investigate the day 1 1996 negative value, but assume it is a small rounding error from the interpolation. The difference there was also set to be 0 instead of a negative value. For Oahu, the warning was just present for Jan 1 1996.
* Climatologies for cumulative variables used mean values multiplied by 365 days, so that they could be directly compared across years. This does mean that leap years are short one day’s accumulation for cumulative variables.
* Quality Control Flag: UDROFF values should be checked for the correct orders of magnitude. In particular, one year (1998) was an order of magnitude higher on average than the other years for Kauai present. Is this plausible? Does this variable have some extreme variation?
* QC outputs are missing for T2 variable (except for Kauai present). This can be an exercise for the next user to test that they can successfully use the processing scripts.

## Overview of HI\_WRF GitHub Repository and Scripts

**Folders:**

3D: Two scripts for processing the 3D data set into cloud water interception. Note that these require the ten 16 TB hard drives that have the WRF simulation output on it in order to use. The ‘for\_points’ version was written to work with 5 locations with field measurements, while the for\_island version is intended to create island-wide calculations. **NOTE: the calculations have not been finalized yet, so please treat all outputs as preliminary unless further validation/vetting is performed.**

gis/Vector: Public Domain Vector format outlines for each of the Hawaiian Islands downloaded from Natural Earth.

grids/templates: Raster templates from the Hawaii Rainfall Atlas to use to convert the WRF simulation to a standardized raster grid. Note that this involves downscaling lower-resolution data to higher resolution, and many GIS experts recommend going the opposite direction. This was done to take advantage of the standardized grid of the Hawaii Rainfall Atlas and to simplify data comparisons and use.

grids/wrf\_grids: The Lat/Lon points of each WRF model point, along with the row and column indices in the data matrix. These files can be used to look up real-world locations in the WRF simulation, or can be used to geo-locate WRF model outputs by joining them to the grid.

settings/Settings.R and /PrecipSettings.R: Some basic settings for running the Workflow scripts to keep the Workflow scripts tidy. Basically, if you find an undefined object in the Workflow script, chances are it was defined in the settings file. PrecipSettings.R should be used with Workflow\_ppt.R and Settings.R should be used with everything else.

workflows/Workflow.R This is the main script for processing the 2D data. It will call other scripts in the required order. It is best run interactively with one section at a time, as sometimes a step will result in error or may take some time for processing, in which case it may need to be re-run or corrected before attempting the next step. See 2D Data below for details on the scripts used by this workflow.

Workflows/Workflow\_Cumulative.R: A workflow for non-bucket variables where values are given cumulatively over the simulation.

Workflows/Workflow\_ppt.R: A custom workflow for precipitation. This was used to generate the precipitation data files – **but watch for bugs**. The precipitation scripts were later adapted for processing the temperature data, so it is possible that some functions may have changed since they were originally applied to the precipitation data.

wrf\_tables: VEGPARM.TBL was taken from the WRF simulation (NCAR). See WRF\_License.txt for link to WRF and for license terms (public domain).

**Files:**

Variables.R: This script provides an overview of the 2D variables in the WRF file available for use.

Workflow\_hlpr.R This script contains the functions used by the other scripts. It is best navigated by searching for the desired function, as the functions are present in no particular order, and I navigate it using the search option.

## 2D Data

To download the precipitation data, the workflow is laid out in Workflow\_ppt.R.

The workflow will need to be run separately for each island.

The workflow consists of the following major tasks:

Load Functions: (Workflow\_hlpr.R), loads functions into memory but does not perform any actions

Load Settings (e.g., 000b\_PrecipSettings.R) Runs a settings file, , which contains instructions on how to run the rest of the processing. This will load variables into the active session.

Make a grid (was only run once, no need to run again) #\*\*# Pull out into a separate script then!

Download the chosen variable(s) for each scenario (0000\_Data\_Downloader.R). Downloads the data for an island. The speed of this script will depend on your internet connection, and may take several hours per island and scenario. The resulting data download may be large (10’s of GB)

#\*\*# DO WE NEED AN ANNUAL HOURLY FILE? #\*\*#

Process the downloaded hourly data into a daily file (001c\_ExtractAnnual\_hm.R, 001c\_ExtractAnnual\_ok.R, note that oahu/Kauai process much slower due to the missing I\_RAIN variable and the need to calculate differences hourly.)

Convert the daily files to .csv format (Daily\_to\_geotif.R)

Convert the .csv format to raster using ArcGIS tool 003c\_CSV\_to\_Rainfall\_Atlas\_Daily.py (add the ArcGIS Toolbox and the script tool should be available)

Process the daily data into aggregates (monthly and annual climatologies) (#\*\*# ProcessAnnual? Needs re-evaluation now that daily files are created differently)

NOTE: Even though it is presented as a single overview script, I prefer to run it in blocks. That is why all the blocks are turned off by default – I typically just run the inside contents of a block in R Studio. (Note that you can set it up to run multiple blocks consecutively. This may be useful in areas where the processing is more robust and less likely to glitch out and crash.

**Example:** Downloading the UDROFF data for Hawaii

**Example:** Processing the UDROFF data for Kauai

Check that the initial paths are correct

make.grid is defined in the settings, it should be set to 0, as the xy grids should already have been generated and are available from the HI\_WRF repository. Skip this.

Skip the Download Data block – for this example the data were already downloaded. If you need help downloading data see the Download Data example above.

**Interpolate Day block:**

This block will add in a missing day (Jan 1, 1996) that was lost due to corruption. It will also add extra hours to the end of the simulation to convert from GMT time to local time (because the data are downloaded in GMT and are being switched to HST).

To fill in the missing day for a cumulative variable, the change in amount is divided by 24 and distributed evenly across the 24 hours. To fill in for a non-cumulative variable, it just repeats the immediate previous day. This is only needed for the present day scenario.

To add data to the end of the simulation, for the cumulative variables (SRFOFF and UDROFF, the values are just fixed at the prior day’s values for the final hours. For non-cumulative variables where a mean is calculated, the extrapolated values are just set to the recent prior values.

Note: this will fill in data in the hourly folder based on the hourly\_raw folder. So if this block is accidentally run twice, it will just overwrite the previously run files, but should not continue to add extra days into the middle of the simulation (it ONLY interpolates that one specific day). You can tell if this script has been run if the hourly folder contains all of the data, ending in 175330 (there will also be the 175320\_deleteme.rda file that contains the interpolated day but not the added extra hours, this can be safely deleted).

**Extract Variables block:**

This will convert the data from hourly to daily data by year. For cumulative variables this will give a total. For non-cumulative variables, several metrics are available #\*\*# FILL IN METRICS.

**Future Directions:**

Some variables were missing from the USGS data server data. At least in the case of UDROFF and SFROFF, these are available in the 3D data set. Scripts could be developed to extract and compile this information, but that is outside the scope of the present project, which was to develop scripts to access the USGS data.

## 3D Data

**Calculate Cloud Water Interception for Selected Points**

In order to calculate cloud water interception, you need to have the 3D WRF data set, (renamed for use, see Section X: File Renaming) for the location and years of interest, you need to have the wind heights extracted using the Python Script (see Section XX Wind Height Extraction Process), and you need the R scripts 3D\_calculation\_v3.R (#\*\*# ADD Dependencies and required packages).

Pre-processing the location file

Once the location file is pre-processed, simply adjust the island group, scenario, and years, and ensure that the input and output drives are correct, then run the script. It was processing a year in ~5 minutes for Oahu and Kauai and in ~25 - 36 minutes for Hawaii/Maui on my PC.

**Calculate Cloud Water Interception**

In the 3D subfolder, there three scripts:

3D\_calculations\_for\_island.R

* This is designed to calculate the cloud-water interception for an entire island from the 3D data set (run on the version adjusted to not contain special characters)

3D\_calculations\_for\_island\_many\_nc\_files.R

* This is designed to calculate the cloud-water interception for an entire island from the extracted 3D variables (on an 8 TB hard drive)

3D\_calculations\_for\_points.R

* This is designed to calculate cloud-water interception for specified points on the Hawaiian islands, using the full 3D data set (run on the version adjusted to not contain special characters).

**File Renaming**

#\*\*# ADD DOCUMENTATION AND TRANSFER ‘script’ FROM LINUX

**Wind Height Extraction Process**

First, check if the wind height has already been extracted. There should be a folder in the 2D data set hard drive labeled wind\_heights. There should be a sub-folder for each island set and scenario. Hawaii/maui are on the one labeled ‘hawaii’ and Oahu and Kauai are on the one labeled ‘kauai\_oahu’. There should be a year subfolder for each year that has been processed.

* At the time of this documentation writing (2023-12-22), 1999-03-16 is missing for hawaii\_present – this file was corrupted on the 16 TB hard drive

If the wind height has not already been extracted, or you are using this set of code on a different WRF model then:

Open the Extract\_wind\_level\_height.py script with your favorite Python IDE. You will need the netCDF4 module and the wrf module. Note that for installation purposes, the wrf module is the wrf-python module. I had some trouble installing this, but finally got it to work on one computer using the conda install command through my Anaconda installation. (pip install was not going well on a different computer outside of Anaconda).

Here is a link to the documentation for the tool for extracting wind heights:

https://wrf-python.readthedocs.io/en/develop/internal\_api/generated/wrf.g\_geoht.get\_height\_agl.html

Because some of the 16 TB hard drives were very full, I set it up so that the wind heights would be extracted to a different drive than the original data hard drives.

You will need to adjust the paths to fit the input hard drive – each one had a slightly different file structure, but the path\_bits object can be customized to contain the part of the path that changes between years.

It is a good idea to check that the length of the year\_list and either the path\_bits or path\_list match. Use either path\_bits or path\_list. path\_bits if you just want to fill in a small part of each path, use path\_list if you just want to write out the paths you will be using. They are basically two different ways of doing the same thing, so it doesn’t really matter which approach one uses.

NOTE: The year folders CANNOT contain any other files, or the script will crash as written. (exceptions could be added to the script if desired, but was not necessary in this case).

Once the paths are adjusted properly, simply run the script, and it should output each day that is processed. This will give the wind layer heights needed for the cloud water interception wind downscaling calculation.

**Future Directions:**

A wind stability correction could be applied to the downscaled wind data. However, this will not solve the concern about WRF wind speeds below the canopy height, as the logarithmic model is assuming 0 wind at the 10 m height for a 35 m canopy. So this would give us a refined wind estimate, but does not resolve the fundamental question about why the WRF wind speeds are not lower within the vegetation canopy. This refinement is probably only worthwhile IF the model outputs are close to the observed outputs. If there is a big gap, it is unlikely that this will matter.

For the Cloud Water Interception project, I would recommend extracting out just the variables of interest from the 3D data set and saving these as a separate WRF file. I think a 90% data reduction could be achieved, which could lead to everything being on a single 16 TB hard drive, instead of on 10. In Python, an xarray object can contain multiple data sets, and there are libraries in Python and C for writing data to netcdf format.

## Troubleshooting

**I got this error or something similar while downloading the data:**

**CURL Error: Failure when receiving data from the peer**

**Error in Rsx\_nc4\_get\_vara\_double: NetCDF: DAP failure**

**Var: RAINNC\_rcp85 Ndims: 3 Start: 17579,0,0 Count: 8785,64,82**

**Error in ncvar\_get\_inner(ncid2use, varid2use, nc$var[[li]]$missval, addOffset, :**

**C function R\_nc4\_get\_vara\_double returned error**

Sometimes it has memory issues while downloading. Find the spot where the download script left off and try to resume there (by running the DataDownloader manually with appropriate start/end numbers. Alternatively, you can try downloading everything again in a new session (perhaps reboot the computer to try to free up any memory that may have been tied up, or try running it with fewer other processes at the same time).

**I am running the downloader script, but it is not doing anything.**

Stop the R process, restart R, check your internet connection, and try again. Likely the internet connection was disrupted, the process hung up. If you are careful, you can adjust the start and end points to run for just the remaining files, instead of re-running for everything.

**There are small negative values in the array when I switch from hourly to daily**

This appears to be a rounding error related to the interpolation. This will be corrected at a later step, or one can take the data out of array, replace with 0’s, and then put the data back into an array format to correct it. (this was not done to avoid getting the order wrong, and there may be an easier tool for this!)

## Acknowledgements