The meteorological downscaling can be run from run_met_downscaling.R (located in FLARE/Rscripts/met_downscale)

This will run all of the steps necessary to:

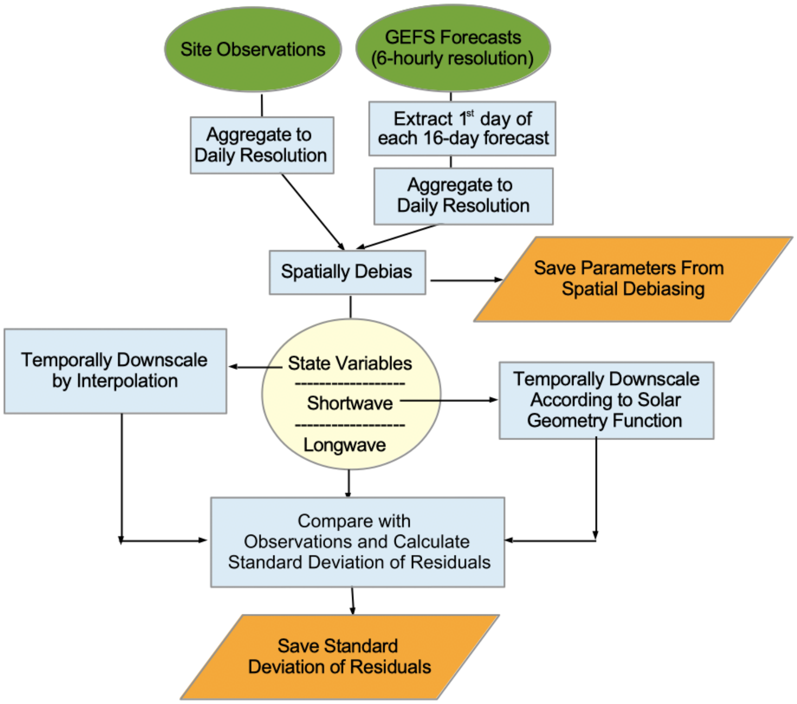
- build linear models from the relationship between site observations and historical GEFS forecasts at the daily resolution (parameters only need to be fit the first time, this option can be turned off for subsequent runs)
- Print and save the coefficients related to these models
- Spatially debias at the daily resolution using the saved coefficients
- Temporally downscale from 6-hourly to hourly using below methods

Meteorological variable	Method of temporal downscaling
Air Temperature at 2m	Interpolation
Relative Humidity at 2m	Interpolation
Wind Speed at 10m	Interpolation
Shortwave Radiation Flux	Solar Geometry
Longwave Radiation Flux	None

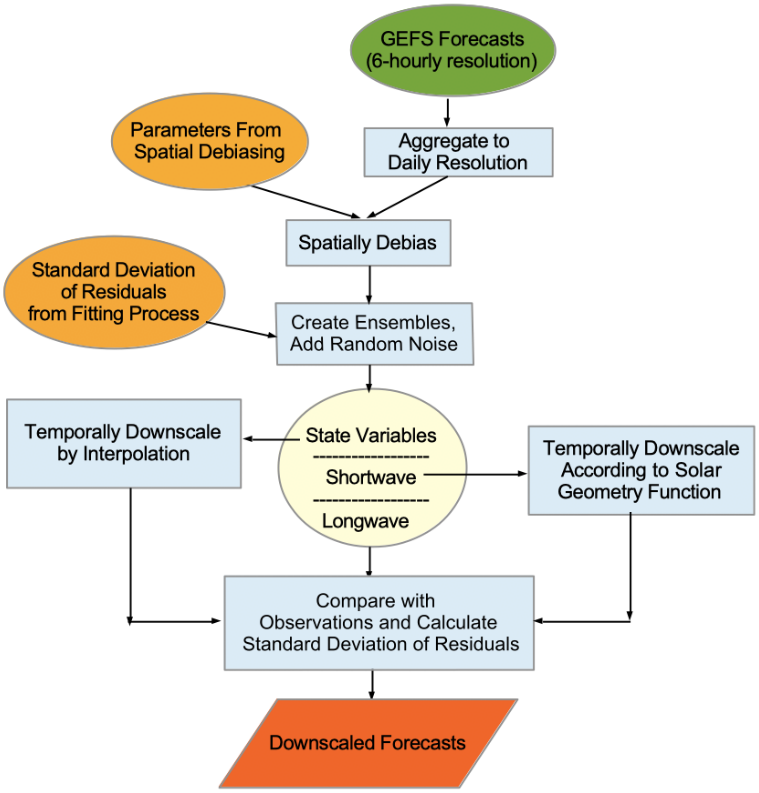
- Build a covariance matrix between related variables
- Create ensemble members and add random noise from the covariate noise distribution
- Produce plots comparing the output to observations (only use this for validation)

Visual representations of the “parameter-fitting” (really just getting coeffients) and downscaling processes are on the following page

Parameter Fitting Process


Site Observations
GEFS Forecasts (6-hourly resolution)
Aggregate to Daily Resolution
Extract 1st day of each 16-day forecast
Aggregate to Daily Resolution
Spatially Debias
Save Parameters From Spatial Debiasing
Temporally Downscale by Interpolation
State Variables
Shortwave
Longwave
Temporally Downscale According to Solar Geometry Function
Compare with Observations and Calculate Standard Deviation of Residuals
Save Standard Deviation of Residuals

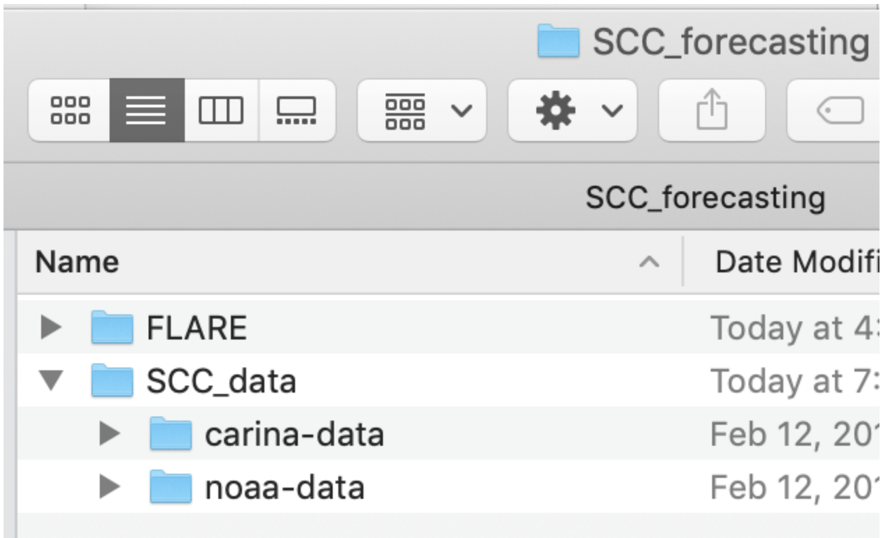
Downscaling Process


GEFS Forecasts (6-hourly resolution)
Parameters From Spatial Debiasing
Aggregate to Daily Resolution
Spatially Debias
Standard Deviation of Residuals from Fitting Process
Create Ensembles, Add Random Noise
Temporally Downscale by Interpolation
State Variables
Shortwave
Longwave
Temporally Downscale According to Solar Geometry Function
Compare with Observations and Calculate Standard Deviation of Residuals
Downscaled Forecasts

To get started: download files and set up file structure

Create an empty folder called SCC_forecasting and put inside of it:

- FLARE (download folder from https://github.com/CareyLabVT/FLARE)
- SCC_data
    - carina_data (download folder from https://github.com/CareyLabVT/SCCData/tree/carina-data)
    - noaa-data (download folder from https://github.com/CareyLabVT/SCCData/tree/noaa-data



Name	Date Modifi
▶ FLARE	Today at 4
▼ SCC_data	Today at 7
▶ carina-data	Feb 12, 20
▶ noaa-data	Feb 12, 20

How to run with Falling Creek Reservoir Data

- Open **run_met_downscaling.R** (located in FLARE/Rscripts/met_downscaling/)
- Input your directory for the SCC_forecasting folder on line 3
  Now the script should run
- Adjust parameters to produce different output as follows:

  FIT_PARAMETERS
    - If TRUE, observational data and historical forecasts are compared to calculate the linear model coefficients for each variable. These are printed to the screen, saved, and used again if DOWNSCALE_MET is true
    - If FALSE, then the code will look for saved coefficients in if DOWNSCALE_MET is true

  DOWNSCALE_MET
    - If TRUE, forecast is downscaled
    - If FALSE, “out-of-box” scenario is output. This is just an hourly interpolation of the forecasts as is.

met_downscale_uncertainty

- If TRUE, ensemble members with random noise addition are created
- If FALSE, no noise is added

Compare_output_to_obs

- If TRUE, the output is compared against observations and the results are printed

file_name = "20190101gep_all_00z"

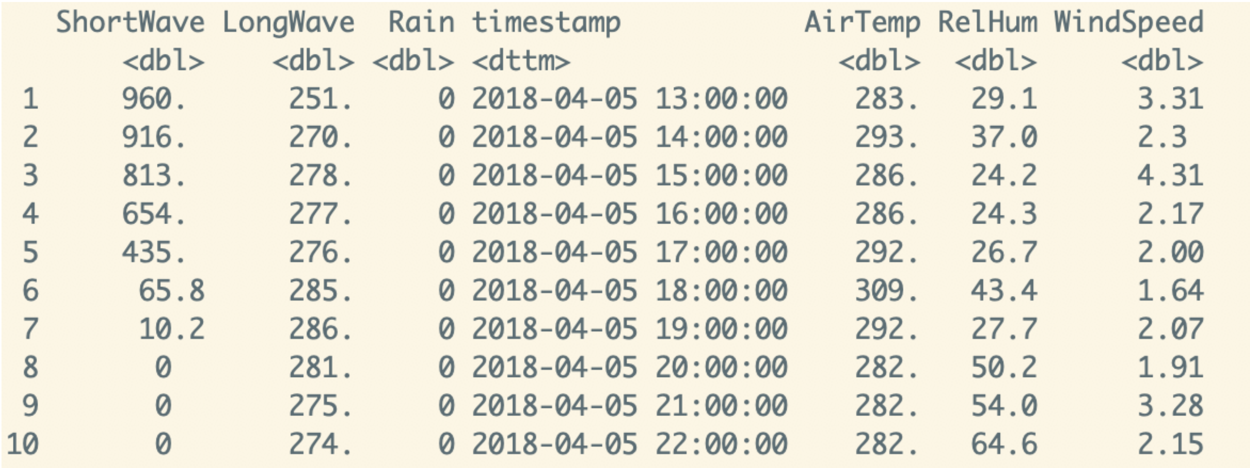
- The name of the forecast file to be downscaled. These are stored in the noaa-data folder

n_ds_members

- The number of downscaling ensemble members (so if there are 21 input noaa GEFS ensemble members, and n_ds_members = 10, then the output would have 210 ensemble members)

How to apply this process to different sites
(this hasn't been done yet, and will require some debugging)

- Get the observational data into the same format as used in the example above. To do this, I would recommend removing lines 43-78 of process_downscale_GEFS.R (stored in FLARE/Rscripts/met_downscaling) and replacing the hrly.obs object with your own dataframe. Column order does not matter, but "timestamp" must be a column of hourly times formatted as datetime objects. Example dataset shown below.



```
   ShortWave LongWave  Rain timestamp           AirTemp RelHum WindSpeed
       <dbl>    <dbl> <dbl> <dttm>                <dbl>  <dbl>     <dbl>
 1     960.      251.     0 2018-04-05 13:00:00    283.   29.1      3.31
 2     916.      270.     0 2018-04-05 14:00:00    293.   37.0      2.3 
 3     813.      278.     0 2018-04-05 15:00:00    286.   24.2      4.31
 4     654.      277.     0 2018-04-05 16:00:00    286.   24.3      2.17
 5     435.      276.     0 2018-04-05 17:00:00    292.   26.7      2.00
 6      65.8     285.     0 2018-04-05 18:00:00    309.   43.4      1.64
 7      10.2     286.     0 2018-04-05 19:00:00    292.   27.7      2.07
 8       0       281.     0 2018-04-05 20:00:00    282.   50.2      1.91
 9       0       275.     0 2018-04-05 21:00:00    282.   54.0      3.28
10       0       274.     0 2018-04-05 22:00:00    282.   64.6      2.15
```

- Update "**VarInfo**" in **main_met_downscaling.R** to reflect the variables that you will be downscaling
  (If you are using different variables, there are other parts of the code that will have to be changed. About half of the process is generalized by using "VarInfo" input, but variables names are still hard-coded into some of the process.)

- Update **first_obs_date** and **last_obs_date** to be the dates of the first and last days of historical GEFS forecasts used to build relationship with observational data

- In **process_downscale_GEFS.R** (stored in FLARE/Rscripts/met_downscaling) update **Line 50, and 51**: (set the min and max bounds for air temperature)

- Update other relevant input parameters in run_met_downscaling.R