**Ryan's Python Scripts:**

**####### Plot Scripts #######**

* ***plot\_matrix\_calib\_error\_3bias\_subplots\_CHPS.py***: create imshow matrix figure with 3 subplots. Input data is the .csv SQME/QME file directly from CHPS (check for proper column reference). View CHPS simulation/observed data in plot table, set view period and save as .csv. Features: format color bar, imshow, matrix, fontsize changes, tick locations. [D:\Projects\NWS\Python\plot\_CRAT](file:///D:\Projects\NWS\Python\plot_CRAT)
* ***plot\_RFC\_PET\_Initial.py***: create line subplots of all basin ET demand curve data (apriori, initial sa, FAO P&M) from .csv files. Must be modified for each RFC depending on the number of basins. [D:\Projects\NWS\Python\plot\_PET](file:///D:\Projects\NWS\Python\plot_PET)
* ***plot\_UHG\_compare.py***: create line plot of before and after calibration for each basin UH ordinates using UH data from extract\_hydro\_params script output .csv file [D:\Projects\NWS\Python\UNIT\_HG](file:///D:\Projects\NWS\Python\UNIT_HG)
* ***plot\_PET\_compare.py***: create a annual time series line plot to compare several ET-demand mid-month values for a basin. Each time series source is located in an individual .csv file and output is a plot for each basin. Rainbow plot. [D:\Projects\NWS\Python\PET](file:///D:\Projects\NWS\Python\PET)
* ***plot\_daily\_correlation\_XXRFC.py***: plots a 1 to 1 correlation plot of observed vs. simulated daily streamflow. A polynomial trend line is fitted to the data and the equation of the line is displayed on the plot. Also outputs a csv file with calculated % bias within a range of defined bins [D:\Projects\NWS\Python\plot\_daily\_correlation](file:///D:\Projects\NWS\Python\plot_daily_correlation) (uses deepcopy to copy dictionary and collections to maintain dictionary order)
* ***plot \_UH\_analysis.py:***This script plots the mean % bias for the period surrounding an event of a specified magnitude. Numerous events are chosen at each basin based on the criteria that eliminate smaller events and events that may be impacted by multiple events. Updated 8/11/2015 to plot mean event time series as second subplot (MBRFC FY15) Key words: appending list to matrix, matrix and array calculations, modify xticks, add line at x origin with autoscale off: 0. [D:\Projects\NWS\Python\UNIT\_HG](file:///D:\Projects\NWS\Python\UNIT_HG)
* ***plot \_SNODAS\_ADC.py:***This script plots a scatter plot of the NOHRSC SNODAS modeled SWE and SCA  using downloaded data from NOHRSC website. Features: second twin axis with different units, join two lists to a dictionary, pandas read csv. [D:\Projects\NWS\Python\SNOW](file:///D:\Projects\NWS\Python\SNOW)
* ***plot\_ADC\_compare.py:*** script plots pre- and post-calibration ADCs. [D:\Projects\NWS\Python\SNOW](file:///D:\Projects\NWS\Python\SNOW)
* ***plot\_param\_ranges.py:*** script to plot SAC/Snow param ranges using extracted moduleparfile csv files. Data plotted: pre-calb, draft calb, final calb, Anderson, apriori. key features: pandas dataframes, axis tick labels inside plot, annotate with mask/shadow around text, watermark logo inside figure [D:\Projects\NWS\Python\Extract\_Hydro\_Params](file:///D:\Projects\NWS\Python\Extract_Hydro_Params)
* ***plot\_Tatum\_compare.py:*** script to plot changes to Tatum coefficients and layer thresholds (pre calb vs post calb) key features: colormap for automatic line color spectrum [D:\Projects\NWS\Python\UNIT\_HG](file:///D:\Projects\NWS\Python\UNIT_HG)
* ***plot\_station\_data\_timeline.py:***  script creates an horizontal bar chart of station data availability for precip/temp data sites. Good for determining record overlap. [D:\Projects\NWS\Python\Cardfiles](file:///D:\Projects\NWS\Python\Cardfiles)

**####### Interactive Plot Scripts #######**

* ***bokeh\_plot\_streamflow.py***: generate an interactive plot of QME vs. SQME using bokeh module. Output to an html file for viewing. [D:\Projects\NWS\Python\bokeh](file:///D:\Projects\NWS\Python\bokeh) (update bokeh v11: <http://bokeh.pydata.org/en/server_branch/docs/installation.html>)

**####### Geographic Map Plot Scripts #######**

* ***plot\_shapefiles\_serfc.py***: Basemap module. Example plot using ArcGIS background layer with basin shapefiles. [D:\Projects\NWS\Python\basemap\_plots](file:///D:\Projects\NWS\Python\basemap_plots)

**###### Data Download ########**

* ***QME\_NWS\_datacard\_download.py:*** loop through basin summary csv files and download USGS QME datacard files from <http://dipper.nws.noaa.gov/hdsb/data/archived/usgs.html>. Uses urllib2 module. [D:\Projects\NWS\Python\QME](file:///D:\Projects\NWS\Python\QME)
* ***QIN\_USGS\_csv\_download.py:*** loop through basin summay csv files and download USGS QIN data files from historic data site (pre WY 2007) <http://ida.water.usgs.gov/ida/> and recent site (2007 WY+) <http://waterdata.usgs.gov/nwis>. Uses mechanize module to automate webpage submission and retrievals. [D:\Projects\NWS\Python\QIN](file:///D:\Projects\NWS\Python\QIN)
  + ***merge\_USGS\_QIN\_timeseries\_csv.py:***creates a QIN time series in .csv format for import into CHPS. Capable of using the 2 files from the USGS -> downloaded with QIN\_USGS\_datacard\_harves.py: 1. historical (before Oct 1, 2007) txt file 2. recent (after Oct 1, 2007) txt file. Data retrieval info: <http://waterdata.usgs.gov/nwis/?IV_data_availability.> USGS parameter codes: <http://nwis.waterdata.usgs.gov/usa/nwis/pmcodes>
* ***download\_basin\_snow\_climo.py:*** Download NOHRSC SNODAS data from the NOHRSC basin plot website. Input a list of basins and output hourly snow data to a csv. [D:\Projects\NWS\Python\NOHRSC](file:///D:\Projects\NWS\Python\NOHRSC)

**####### Parse Files #######**

MAP/MAT/PET/Streamflow

* ***map\_mat\_format\_conversion.py***: input raw 6 column map/mat data and create .txt file in single column with datetime object [D:\Projects\NWS\Python\MAP\_MAT](file:///D:\Projects\NWS\Python\MAP_MAT)
  + 9/2/2014: modified script to assign the first value of the month as the "Day 1 06:00:00" observation instead of "Day 1 00:00:00" observation -> matches CHPS data
* ***map\_water\_year\_analysis.py****:* input 2 column (datetime, MAP) file from map\_mat\_conversion.py above and create a csv with water year totals [D:\Projects\NWS\Python\MAP\_MAT](file:///D:\Projects\NWS\Python\MAP_MAT)
  + ***map\_monthly\_normal\_analysis.py***: same format as above except calculate on a mean monthly basis for a user define period [D:\Projects\NWS\Python\MAP\_MAT](file:///D:\Projects\NWS\Python\MAP_MAT)
* ***PRISM\_summary\_table.py***: reads a list of .xls (or .csv) files output from the PRISM arcgis tool or python arcpy script (1 file for each basin) and call a specific cell value in .xls -> output a summary .csv file [D:\Projects\NWS\Python\PRISM](file:///D:\Projects\NWS\Python\PRISM)
  + ***PRISM\_summary\_table\_monthly.py***: reads a list of  .csv files output from the PRISM arcgis tool or python arcpy script (1 file for each basin for each month) and calls the monthly precip, tmin, tmax, or tmean value (updated 10/14/2014). Converts precip from mm to inches and outputs a summary .csv file with all basins. [D:\Projects\NWS\Python\PRISM](file:///D:\Projects\NWS\Python\PRISM)
* ***QME\_summary\_statistics.py***: input USGS discharge text files or CHPS csv exported QME and create csv with summary statistics (mean, max, min, sd) and date of max/min (masked array handling). Also generates a horizontal bar plot showing the timeline of available QME data.  [D:\Projects\NWS\Python\QME](file:///D:\Projects\NWS\Python\QME)
* ***QIN\_summary\_statistics.py***: This script reads CHPS csv file from QIN plot display or raw USGS data (created by \_create\_QIN\_timeseries.py) and finds the start and end of the observed hourly QIN record, # of valid data points, and % of total available. Outputs summary data to csv file. [D:\Projects\NWS\Python\QIN](file:///D:\Projects\NWS\Python\QIN)
  + ***daily\_to\_annual\_conversion\_xls.py***: parse a spreadsheet with date/time and daily discharge values and calculate a mean daily discharge for each water year for water balance
* ***QME\_water\_year\_statistics.py***: calculates a WY summary of QME data from NWS datacard data and outputs a water year mean daily flow table [D:\Projects\NWS\Python\QME](file:///D:\Projects\NWS\Python\QME)
* ***QME\_datacard\_to\_csv.py***: converts individual QME datacard files to a single/merged csv file that can be imported for dss build [D:\Projects\NWS\Python\QME](file:///D:\Projects\NWS\Python\QME)
* ***parse\_apriori\_pet.py***: parse through the apriori .csv file from arcpy script and pull the PE, PEadj, and ETD data and write to a new separate .csv file for each basin [D:\Projects\NWS\Python\APriori](file:///D:\Projects\NWS\Python\APriori)
* ***xls\_to\_csv.py***: convert .xls and .xlsx files within a directory to .csv files [D:\Projects\NWS\Python](file:///D:\Projects\NWS\Python)
* ***daily\_to\_annual\_resdata\_xls.py***: parse through .xlsx files containing daily reservoir storage and outflow data. Create a summary table (.csv file) based on water year analysis [D:\Projects\NWS\Water Balance\LMRFC\LMRFC\_FY2014](file:///D:\Projects\NWS\Water%20Balance\LMRFC\LMRFC_FY2014)
  + ***daily\_to\_annual\_resdata\_txt.py***: same script but designed to use .txt file with inflow, outflow, pumpback variables
  + ***daily\_to\_annual\_resdata\_chps\_import.py***: same script but designed to use .txt files used in CHPS import. Data is already in cfsd
* ***mat\_month\_mean\_daily\_min\_max.py***: parse through column MAT data (from map\_mat\_conversion.py) and calculate a monthly mean daily max and min temperatures for use in PET calculations -> output to csv file for each year (row) [D:\Projects\NWS\Python\MAP\_MAT](file:///D:\Projects\NWS\Python\MAP_MAT)
* ***mape\_day\_to\_month\_climo.py***: Calculates a climatology mean monthly PE timeseries for a basin using an exported .csv time series containing daily data. Outputs a .csv with monthly data. [D:\Projects\NWS\Python\MAPE](file:///D:\Projects\NWS\Python\MAPE)
* ***\_create\_QIN\_timeseries.py***: Cody's script to parse QIN historical and recent USGS download files. Create a single basin QIN .csv format ready for CHPS import [D:\Projects\NWS\Python\QIN](file:///D:\Projects\NWS\Python\QIN)

Prec & Temp Processing

* ***parse\_iem\_tocardfile.py***: parse through individual data files from IEM website (e.g. hourly ASOS/AWOS) and generate formatted cardfile. Also creates a summary csv file with calculated valid data points and percent of total. Summary file used to display available data stats in arcmap. [D:\Projects\NWS\Python\Cardfiles](file:///D:\Projects\NWS\Python\Cardfiles)
* ***parse\_nhds\_tocardfile\_plot.py***: parse through a summary file of NHDS site info obtained from website and split out individual cardfiles for each site. Also creates a summary csv file with calculated valid data points and percent of total. Summary file used to display available data stats in arcmap. Also has option to create station summary bar plot of available data. [D:\Projects\NWS\Python\Cardfiles](file:///D:\Projects\NWS\Python\Cardfiles)
* ***parse\_raws\_tocardfile\_plot.py***: parse through a individual RAWS data files (Stitches together data prior to 2004->WRCC RAWS and post 2004 data->RAWS site) and generate formatted cardfile. Also creates a summary csv file with calculated valid data points and percent of total. Summary file used to display available data stats in arcmap. Also has option to create station summary bar plot of available data. Key Features: day of year coversion to date, padded zero integer format.  APRFC 2015 task. [D:\Projects\NWS\Python\Cardfiles](file:///D:\Projects\NWS\Python\Cardfiles)
* ***parse\_scan\_tocardfile.py***: parse through a individual data files from SCAN website and generate formatted cardfiles. Also creates a summary csv file with calculated valid data points and percent of total. Summary file used to display available data stats in arcmap. [D:\Projects\NWS\Python\Cardfiles](file:///D:\Projects\NWS\Python\Cardfiles)
* ***pxpp\_create\_incard.py***: generate an input file with the pxpp station data using the summary file and a list of the cardfiles. [D:\Projects\NWS\Python\NSWRFS\_preprocessors](file:///D:\Projects\NWS\Python\NSWRFS_preprocessors)
* ***mat\_create\_incard.py***: generate an input file with for the MAT preprocessor using station history csv and MAT consistency check output files. MAT input [format](http://www.nws.noaa.gov/oh/hrl/nwsrfs/users_manual/part3/_pdf/38mat.pdf). [D:\Projects\NWS\Python\NSWRFS\_preprocessors](file:///D:\Projects\NWS\Python\NSWRFS_preprocessors)
* ***map\_create\_incard.py***: generate an input file with for the MAP preprocessor using station history csv and pxpp output files. MAP input [format](http://www.nws.noaa.gov/oh/hrl/nwsrfs/users_manual/part3/_pdf/37map.pdf). [D:\Projects\NWS\Python\NSWRFS\_preprocessors](file:///D:\Projects\NWS\Python\NSWRFS_preprocessors)
* ***parse\_conagua\_tocardfile\_plot.py***: (daily and hourly scripts) parse through a individual CONAGUA csv files to cardfile. [D:\Projects\NWS\Python\Cardfiles](file:///D:\Projects\NWS\Python\Cardfiles)
* ***pxpp\_annual\_precip\_results.py:*** parse through the PXPP output results summary file and locate the station id and the calculated annual precip value. Output to a csv file for analysis. [D:\Projects\NWS\Python\NSWRFS\_preprocessors](file:///D:\Projects\NWS\Python\NSWRFS_preprocessors)

CHPS Realated

* ***extract\_hydro\_routing\_params\_sa.py:*** same as above, but using the original chps SA moduleparfiles (.xml) directly from the CHPS SA/Calb config-> different file format. Updated name and features to plot the lag/k variables. [D:\Projects\NWS\Python\Extract\_Hydro\_Params](file:///D:\Projects\NWS\Python\Extract_Hydro_Params)
* ***extract\_hydro\_params\_calb\_slim.py:*** extracts SAC-SMA/UNITHG/LAG-K parameters values from CHPS configuration .xml files located in the Config/ModuleConfigFiles directory and ouputs a .csv file with all parameters  [D:\Projects\NWS\Python\Extract\_Hydro\_Params](file:///D:\Projects\NWS\Python\Extract_Hydro_Params)
* ***statqme\_summary\_table\_create\_csv.py***: parse the CHPS output statqme html reports (beautiful soup). Create summary tables of monthly data (pbias, bias, rms), simulation fit (correlation coef, daily rms error, daily abs error), and flow interval stats. Output data to csv format for report. [D:\Projects\NWS\Python\STATQME](file:///D:\Projects\NWS\Python\STATQME)
* ***pixml\_to\_cardfiles.py***: read through large pixml\_mapx file and output hourly data in OHD cardfile format. ##Card files not tested in CHPS!! Key features: ordered dictionaries, strptime module to parse dates, formatted float output. [D:\Projects\NWS\Python\Cardfiles](file:///D:\Projects\NWS\Python\Cardfiles)
* ***csv\_to\_datacard.py***: read through csv file and output data in OHD single column cardfile format. Note: script does not check for missing time steps. Key features: ordered dictionaries, strptime module to parse dates, formatted float output. [D:\Projects\NWS\Python\Cardfiles](file:///D:\Projects\NWS\Python\Cardfiles)
* ***parse\_chps\_QIN\_summary.py***: reads CHPS csv file from QIN plot display and finds the start and end of the observed hourly QIN record, # of valid data points, and % of total available. Summary data is output to a .csv file. Features: pandas, masked list, navigate back directory. [D:\Projects\NWS\Python\QIN](file:///D:\Projects\NWS\Python\QIN)
* ***qin\_sqin\_statistics***: reads CHPS csv file from QIN plot display and calculates error statistics using calc\_errors.py module. Key features: calls a module and function from an outside directory. [D:\Projects\NWS\Python\QIN](file:///D:\Projects\NWS\Python\QIN)
* ***round\_UHG\_oridinates.py:*** parse through UHG .xml parameter files from exported CHPS mods and create a new file with ordinates rounded to whole numbers (CHPS version currently leaves multiple decimals). [D:\Projects\NWS\Python\plot\_UHG](file:///D:\Projects\NWS\Python\plot_UHG)
* ***update\_coldstates.py:***create a new ColdStateFiles directory by copying the old directory and replace the zip file with a new file containing the moduleparfile from from the exported calibration mod and a copy of the original statesI.txt Key features: extracts file from zip file, creates new zip file, checks if directory exists, check if file exists, copy directory tree. [D:\Projects\NWS\Python\CHPS](file:///D:\Projects\NWS\Python\CHPS)
* ***refresh\_coldstates\_parfiles.py:*** Within CHPS config create a new ColdStateFiles directory by copying the old directory contents  and replacing the params\_previous.xml with a new file using the moduleparfile. Also copy the original statesI.txt to the new directory . The script also renames the original ColdStateFiles directory to "ColdStateFiles\_previous"  and renames the new directory to "ColdStateFiles". [D:\Projects\NWS\Python\CHPS](file:///D:\Projects\NWS\Python\CHPS)
  + ***Config\_refresh\_coldstates\_parfiles\_bybasin.py:*** update the ColdStateFiles in a config directory using the parameter files in the config ModuleParFile directory. Creates a new "updated\_ColdStateFiles" directory to examine for replacing the original. Execute from the main Config directory
* ***UHG\_xml\_param\_updates.py:***Loop through calibration basin UNIT-HG ModuleParFiles and update the UHG ordinates and area values using excel generated updated values. Input a csv with the updated zone area and UHG ordinates. [D:\Projects\NWS\Python\UNIT\_HG](file:///D:\Projects\NWS\Python\UNIT_HG)
* ***config\_update\_modparfiles.py:*** copy/replace any updated ModuleParFiles to a config ModuleParFiles directory

**####### Arcpy scripts #######**

* ***extract\_basin\_apriori\_grid\_values.py***: arcpy tools to extract raster by basin shapefile convert to points and output .txt for each SACSMA attributes (also finds a list of basins from file names in director by ignoring non-directory files [D:\Projects\NWS\GIS\python](file:///D:\Projects\NWS\GIS\python)
  + modified to also create a single summary csv file for each basin using all of the individual txt files
* ***extract\_basin\_DEM\_statistics.py***: arcpy tools to extract elevation and slope summary for multiple basin shapefiles and write to a .csv file (using csv write) [D:\Projects\NWS\GIS\python](file:///D:\Projects\NWS\GIS\python)
  + run the merge\_elevation\_slope\_summary.py script to create a single csv file with all of the basins elevation and slope info and convert units from cm to ft [D:\Projects\NWS\Python\Elevation\_Slope](file:///D:\Projects\NWS\Python\Elevation_Slope)
* ***extract\_basin\_prism\_values.py***: arcpy tools to extract prism data from raster for multiple basin shapefiles and write output to separate .csv file for each basin [D:\Projects\NWS\GIS\python](file:///D:\Projects\NWS\GIS\python)
  + run the PRISM\_summary\_table.py to create a single .csv summary file (precip in inches) containing all basins in specified RFC [D:\Projects\NWS\Python\PRISM](file:///D:\Projects\NWS\Python\PRISM)
  + ***extract\_basin\_prism\_monthly\_values.py***: extract monthly temperature data from multiple years of monthly PRISM data (multiple files) and create an output csv file for each basin [D:\Projects\NWS\GIS\python](file:///D:\Projects\NWS\GIS\python)
  + ***extract\_basin\_prism\_normal\_monthly\_values.py***: extract monthly temperature or precip data from the monthly 1981-2010 normal PRISM data (12 files on Q: drive) and create an output csv file for each basin [D:\Projects\NWS\GIS\python](file:///D:\Projects\NWS\GIS\python)
* ***extract\_basin\_nlcd\_grid\_count.py***: arcpy tools to extract nlcd pixel counts for each basin polygon and create a summary table (join attributes) for each basin [D:\Projects\NWS\GIS\python](file:///D:\Projects\NWS\GIS\python)
  + next run the \_calculate\_basin\_summary.py to create a single .csv summary file containing the data for all the basins available [D:\Projects\NWS\Python\NLCD](file:///D:\Projects\NWS\Python\NLCD)
* ***extract\_basin\_aquifer\_recharge \_values.py***: script uses arcpy tools to extract aquifer recharge estimates raster ([D:\GIS Library\rech48grd](file:///D:\GIS%20Library\rech48grd)) by individual basin polygons and calculates basin mean value. IMPORTANT: script fails to loop through a list of basins (server too slow?) -> run each basin one at a time and script will append each basin value to existing csv file. [D:\Projects\NWS\GIS\python](file:///D:\Projects\NWS\GIS\python) . Updated 10/31/2014 to use the USGS transmissivity raster for FL, and southern GA.
* ***extract\_basin\_gSSURGO\_data.py***: script to extract gridded SSURO data using individual basin shapefiles. Refer to the README doc or script header for more info and setting up extraction. [D:\Projects\NWS\GIS\python](file:///D:\Projects\NWS\GIS\python)
  + README: [D:\Projects\NWS\GIS\python](file:///D:\Projects\NWS\GIS\python) -> \_README\_gSSURGO\_Method.docx
* ***automate\_param\_maps\_RFC.py***: automate the export map process for all SAC par shapefiles. Use the updatelayer function (requires a layer template for each par) to set the layer properties in the .mxd. Seperate template for the SA parameters and new calib pars. Outputs a .png file. SERFC did not display correctly for some pars on 10.1 (works on 10.2). Updated to use arcpy.labelClasses.  [D:/Projects/NWS/Python/arcpy](file:///D:\Projects\NWS\Python\arcpy) -> specific to each RFC task
* ***extract\_basin\_AEC\_histogram.py***: clip DEM raster to basin/elev zone and convert the resulting attribute table to a .xls output. Script then calculates the percentile values for the area elevation curve and checks for cell values outside the elevation zone splits-> output to chps format txt files. Key features: read .xls, numpy percentile. [D:\Projects\NWS\GIS\python](file:///D:\Projects\NWS\GIS\python)
* ***extract\_polygon\_vertices.py***: use searchcursor to loop through multi polygon feature and extract a set number of vertex coordinates. Coordinates and feature properties are then exported to a summary txt file. For use with MAP preprocessor. [D:\Projects\NWS\Python\arcpy](file:///D:\Projects\NWS\Python\arcpy)

**####### Modules #######**

* ***calc\_errors.py***: calculates error statistics between two variables (obs and model) including: percent bias, mean absolute error, nash sutcliffe, root mean squared error, and correlation coefficient. [D:\Projects\NWS\Python\modules](file:///D:\Projects\NWS\Python\modules) & [D:\Projects\NWS\Calibration\_NWS\python\modules](file:///D:\Projects\NWS\Calibration_NWS\python\modules)
* ***conversions.py***: [D:\Projects\NWS\Python\modules](file:///D:\Projects\NWS\Python\modules) common conversion calculations including:
  + dms\_to\_dd: degrees/minutes/seconds to decimal degrees.
  + strip\_accents: removes accents on text data (used for CONAGUA Mexico precip data)