**Workflow for BOR/USACE Project 2, Task 1:**

Draft version current as of 08/19/2013

Authors: Andrew Newman and Kevin Sampson

Add directory structure here:

**Data generation:**

*Selection of gages:*

MATLAB script: gages\_2\_subset.m

Uses USGS GAGES-II shapefiles that describe every gauge in the GAGES-II dataset. Finds all gages that are part of the HCDN-2009 subset of GAGES-II. GAGES-II input files found at: *http://water.usgs.gov/GIS/metadata/usgswrd/XML/gagesII\_Sept2011.xml*

Outputs a text file listing all the USGS gage IDs and HUC level 02 regions (*hru\_pois\_all.txt*).

Outputs a text file with some basic descriptors of each gage.

*Daymet forcing data:*

**1) Shapefile generation for areal weighted calculations**

Pass *hru\_pois\_all.txt* to Kevin Sampson. Kevin then runs a python script that runs through each gage and grabs the appropriate HRUs from Roland Viger's PRMS HRU geospatial databases. Roland has developed ArcGIS code to grab the correct HRUs for a given gage ID (point of interest, POI). Kevin then uses ArcGIS to simplify the polygon geometry for processing through the USGS GeoData Portal (GDP). This script outputs a shapefile for each HUC-02 region containing the lump and HRU polygons for each gage. Kevin has another script that will merge the lump polygon with 90 m SRTM elevation data to generate elevation bands for each basin. This comes as another shapefile for each HUC-02 region.

Total output: There are two zipped shapefiles for each HUC-02 region. One contains the lump and HRU polygons. The second has the elevation band geometry.

Kevin has further developed python scripts that will submit shapefiles to the USGS GDP and grab the output. These scripts require python, pyGDP (USGS python package), which requires several other python packages: libsxlt1-dev, then OWSLib and lxml.

**2) Submission to the GDP**

To submit a shapefile to the GDP, one needs to change the *indir* and *outdir* variables in the python script *Multiple\_Requstor.py* to specify where the zipped shapefiles reside and where to place the output file from the GDP.

*Multiple\_Requstor.py* then calls *test\_multiple.py* to do the actual submission. Within *test\_multiple.py* the *attribute* variable needs to be modified depending on which type of forcing data you want to generate from the GDP.

For lump forcing data: *attribute='GAGEID'*

For HRU forcing data: *attribute='hru\_id'*

For elevation band forcing data: *attribute='Gage\_Band'*

These scripts are then executed via: *python Multiple\_Requestor.py*

Note that I've only submitted 1-2 zipped shapefiles at a time due to a high failure rate with the GDP and concerns of overloading their server. Reliability has improved and we could likely submit more at one time in the future. That is a moot point going forward with further code development by Kevin to do the areal weighted calculations in house.

The output from *Multiple\_Requestor.py* is a comma delimited text file (CSV) containing all the output variables listed under the *stat* array for every unique attribute value for all seven Daymet variables.

***From Kevin: GIS Processing Steps***

The processing described here is done using Python version 2.7.2 with the ‘arcpy’ module. For much of the processing, ArcGIS version 10.x must be installed on the host computer, allowing use of the ‘arcpy’ module. For the contour\_intervals.py processing, ArcGIS Spatial Analyst Extension is required. Further, processing requires the GeospatialFabric geodatabases for all regions where POIs will be extracted.

Steps:

1. Identify all POIs that will be used, by NHD region. The output should be a .csv file with one column representing the NHD Region and the other column giving the POI ID number.
2. Use the Testifgagesexist.py script, which will read the CSV file provided, and check the appropriate NHD Region GeospatialFabric geodatabase for existence of each POI. The print statements from the script will describe two conditions which are an issue during processing, described below:
   1. The POI does not exist in the appropriate GeospatialFabric geodatabase. If this is the case, this POI may not be used for basin extraction.
   2. More than 1 instance of the POI exists in the GeospatialFabric geodatabase. If this is the case, this POI may not be used for basin extraction because each basin must have a unique POI ID.
3. Use the ‘Multiple\_Requestor\_new.py’ python script, which automates multiple calls to the ‘subsetFeaturesNParams\_limited2.py’ script and associated scripts (param\_prms\_limited.py, param\_prmsAddDefaults.py) developed by Roland Viger in the MoWs group at USGS. This script takes as input the directory for the GeospatialFabric Geodatabase, the region number, and gageID (modified POI ID). This script merges all individual basin shapefiles into one, and simplifies the geometry (200m), then zips the output, which can be directly consumed by the USGS GDP services (web or python). A field is added (GAGEID) which creates a unique and standard gageID field in the shapefile attribute table, which will be used to differentiate individual basins during GDP processing.

NOTE: The output shapefile from ‘subsetFeaturesNParams\_limited2.py’ can be used in conjunction with GDP to create basin weighted grid statistics for each individual basin (by using GAGEID attribute) or all basins together in a bulk fashion (using PROD\_UNIT attribute).

Additional Processing:

Simplify\_Geometry.py – This script will simplify the geometry of all zipped shapefiles (results from ‘subsetFeaturesNParams\_limited2.py’ in a directory.

Contour\_Intervals.py – This script will break up each of the unique basins (based on GAGEID) in each of the output shapefiles from the ‘Multiple\_Requestor\_new.py’ script based on elevation contours. A referenced mosaic dataset was created based on NHDPlusV2 elevation data for the CONUS region, and raster functions were implemented to reclassify to 100m elevation bands (or contour intervals). The script, which requires ArcGIS Spatial Analyst Extension, will create unique polygons for each GAGEID/Elevation band combination. The outputs are simplified to 100m (settled upon after testing multiple simplification thresholds) and zipped for input to GDP services.

**3) Processing of GDP Output**

The MATLAB script *reformat\_cida\_output\_huc\_02\_1980.m* will take the region shapefiles and the CSV (see step 2) file and reformat the output from the GDP into white space delimited text files.

For lump output: A text file named: *%09d\_lump\_cida\_forcing.txt*

The gage ID fits within the 9 digit right justified integer.

For HRU output: Text files for each gage and every HRU within that basin. Named: *%09d\_hru\_%05d\_cida\_forcing.txt* The gage ID fits within the 9 digit right justified integer and the HRU number fits in the 5 digit right justified integer.

For elevation band output: Still needs to be developed. Will follow HRU methodology likely.

*USGS Streamflow data:*

**1) Getting Data from USGS**

Manual Labor: Use <http://waterdata.usgs.gov/nwis/dv/?referred_module=sw>

On the web page, under Site Identifier, select multiple site numbers check-box, then hit submit.

Next, copy all gage IDs for one region, using 8-digit length into search box. Scroll to bottom of page, enter time range under Retrieve data for: fields

Then under output options, select tab-separated data, save to compressed file. Then hit submit and wait to download.

A sample bash script using wget is included in the scripts git repository: *get\_usgs\_streamflow.bash*

This deprecates the stream\_split.bash if downloading gauge data using this script as it downloads one gauge at a time. New wrapper scripts for reformatting the streamflow data will need to be made.

**2) Reformatting output**

Unzip saved file using gunzip (if saved as compressed file).

Run *reformat\_streamflow\_general.awk* on uncompressed file:

*awk -f reformat\_streamflow\_general.awk < uncompressed\_file\_name > output\_file\_name*

Run *stream\_split.bash* on file (in same directory as *output\_file\_name*): *./stream\_split.bash*

Need to manually switch regions and make sure correct gauge IDs are in the bash script.

*NWS Apriori Snow-17/SAC Parameters:*

a) Manual Labor: Get apriori parameters from: [*ftp://hydrology.nws.noaa.gov/pub/parameters/sac\_par/*](ftp://hydrology.nws.noaa.gov/pub/parameters/sac_par/)

Currently using the SSURGO based apriori parameters. See [README File for New SAC-SMA A-priori Parameter Grids](http://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0CCsQFjAA&url=http%3A%2F%2Fwww.nws.noaa.gov%2Foh%2Fhrl%2Fmodelcalibration%2F3.%2520%2520A%2520priori%2520model%2520parameters%2FREADME_for_New_SAC_par_july_2011.doc&ei=WnQSUvmPDuiayAHruoDACA&usg=AFQjCNG56o0lSaJOqWvx9VXhNuzs7JsV4w&sig2=8CCog9oG6KnMAVXhRNbtxQ&bvm=bv.50768961,d.aWc&cad=rja).

b) Run MATLAB script: *hcdn\_nhru\_apriori\_setup.m* This script will generate apriori parameters for each HRU using a simple average of all grid cells that are in the bounding rectangle of the HRU. Output to a text file for each basin for lumped (not used currently) and for each basin HRU for input into HRU calibration portion of Snow-17/SAC calibration code.

File name for lumped: *%09d\_apriori.txt* The gage ID fits within the 9 digit right justified integer.

File name for HRUs: *%09d\_%05d\_apriori.txt* The gage ID fits within the 9 digit right justified integer and the HRU number fits in the 5 digit right justified integer.

**Running Snow-17/SAC:**

**1) Setup**

*Code resides in git repository (see directory structure at top of document).*

Compile code with: *./compile\_sacsnow17\_huc\_02*

*Switching to makefile around 06 September 2013*

This compiles all the Snow-17, SAC, SCE and wrapper code into an executable. It also copies the executable to run directories for each HUC-02 region.

**2) Running shuffled complex evolution (SCE) code with Snow-17/SAC for lumped calibration**

1) DIRECTORY CONTENTS

*cd /d2/anewman/run/13/*

As an example, region 13 contains the following relevant files:

*go\_hcdn\_huc02\_seed.bash*

*go\_huc02\_optimal\_seed.bash*

*test\_hcdn\_huc02.exe*

*namelist.model.huc\_13*

*namelist.model.region\_13*

and a link:

*namelist.model -> namelist.model.region\_13*

The namelist file: *namelist.model.huc\_13* is the base namelist file for this region. The non-optimized Snow-17 parameters are subjectively “tuned” for each region. The non-optimized SAC parameters are the same for all regions.

2) NAMELIST FILE

The namelist file sections are *INIT\_CONTROL, SNOW\_17, SAC\_SMA, SCE*

*&INIT\_CONTROL*

*init\_smois = 150.0, 100.0, 180.0, 100.0, 125.0, 100.0 !*initial soil moisture state for sac-sma

*forcing\_name = "/snowdata/anewman/daymet//01/001195100\_lump\_cida\_forcing.txt"* !GDP areal average forcing file name for gage being processed

*stream\_name = "/d1/anewman/sacsnow17/hcdn\_streamflow/01/001195100\_streamflow.txt"* !USGS streamflow file name for gage being processed

*model\_out = "/d1/anewman/sacsnow17/hcdn\_output/01/001195100\_model\_output.txt"* !Model output file name for gage being processed. Only used when opt = 0

*opt\_name = "/d2/anewman/hcdn\_output/01/region\_01\_opt.txt"* !File name for SCE optimal paramter

sets. Only used when opt = 0.

*gage\_id = 004296000* !9 digit gage ID being run

*dt = 86400*  !model time step in seconds (86400 = 1 day)

*sim\_len = 5475* !simulation length in days for calibration period

*opt = 1*  !opt = 1 runs optimization code, opt = 0 sets code to run model for calibration/validation using optimal SCE parameter sets (opt\_name)

*model = 3*  !not currently used

*start\_calib = 0*  !don't use for production runs. used for internal testing only

*val\_period = 0*  !flag to run validation or not. MUST be 0 when opt = 1

*metric = "rmse"* !can be set to “rmse”, “mse”, or “nse”

*pet\_coef = 1.4,1.26,1.74* !priestly-taylor coefficient for pet calculation (depends on aridity of region 1.26-1.34 for humid grass/forest then up to 1.74 for very dry areas). This parameter is optimized by SCE

*/*

*&SNOW\_17*

*scf = 1.865, 0.1, 5.0* !gauge undercatch correction factor (frozen precip only)

*mfmax = 1.209, 0.8, 3.0* !maximum melt factor

*mfmin = 0.010, 0.01, 0.79* !minimum melt factor

*uadj = 0.043, 0.01, 0.4* !wind adjustment for enhanced flux during rain on snow

*si = 2454.8, 1.0 3500.0* !SWE for 100% SCA

*pxtemp = 1.144, -1.0, 3.0*  !temperature of rain/snow transition

*nmf = 0.15, 0, 0.3* !not optimized, maximum negative melt factor

*tipm = 0.15, 0.01, 1.0* !not optimized, weight of previous time step

*mbase = 0.0, 0.0, 0.3*  !not optimized, temp above which melt occurs

*plwhc = 0.05, 0.0, 0.4* !not optimized, liquid water holding capacity of snowpack

*daygm = 0.03, 0.0, 0.01*  !not optimized, melt rate at base of snowpack

*adc = 0.05,0.15,0.26,0.45,0.5,0.56,0.61,0.65,0.69,0.82,1.0* !not optimized type bc (mountains)

*/*

*&SAC\_SMA*

*uztwm = 21.615, 1.0, 800.0* !upper zone tension water capacity

*uzfwm = 757.617, 1.0, 800.0* !upper zone free water capacity

*lztwm = 541.699, 1.0, 800.0* !lower zone tension water capacity

*lzfpm = 549.209, 1.0, 1000.0* !lower zone primary free water capacity

*lzfsm = 980.741, 1.0, 1000.0* !lower zone supplemental free water capacity

*adimp = 0.0, 0.0, 0.40* !not optimized, additional impervious area

*uzk = 0.698, 0.1, 0.70* !upper zone free water withdrawal rate

*lzpk = 0.001, 0.00001, 0.025* !lower zone primary free water withdrawal rate

*lzsk = 0.045, 0.001, 0.25* !lower zone supplemental free water withdrawal rate

*zperc = 74.264, 1.0, 250.0* !maximum percolation rate coefficient

*rexp = 2.495, 0.0, 6.0* !percolation equation exponent

*pctim = 0.005, 0.0, 0.6* !not optimized, minimum impervious area

*pfree = 0.388, 0.0, 1.0* !fraction of percolated water going directly to lower zone free water storage

*riva = 0.0*  !not optimized, riparian vegetation area

s*ide = 0.0* !not optimized, ratio of deep recharge to channel base flow

*rserv = 0.3* !not optimized, fraction of lower zone free water not transferable to lower zone tension water storage

*unit\_shape = 1.527, 1.0, 5.0*  !gamma distribution shape parameter

*unit\_scale = 2.000, 0.001, 150.0* !scale parameter. limit of 1-200 for a model time step of 24 hours (need dt/max(unit\_scale) > ~0.12)

*/*

*&SCE*

*nopt = 20* ! number of parameters to be optimized

*maxn = 150000* ! maximum number of trials before optimization is terminated

*kstop = 9*  ! number of shuffling loops the value must change by PCENTO (MAX=9)

*pcento = 0.001* ! the percentage

*ngs = 48*  ! number of complexes in the initial population

*npg = 41* ! number of points in each complex (=2\*NOPT + 1)

*nps = 21* ! number of points in a sub-complex (=NOPT + 1)

*nspl = 41* ! number of evolution steps allowed for each complex before shuffling (=2\*NOPT + 1)

*mings = 12* ! minimum number of complexes required (=NGS)

*iniflg = 0* ! 1 = include initial point in the population

*iprint = 1* ! 0 = suppress printing

*iseed = 37* !starting seed for random number generator

*sce\_fname = "/d1/anewman/sacsnow17/sceua\_output.txt"* !Currently not used...

*/*

Parameters that are optimized in *&INIT\_CONTROL, &SNOW\_17, &SAC\_SMA* need three values. Those that are not optimized need one.

Optimized parameters: First value is the starting value, second is the lower bound for SCE, third is the upper bound for SCE.

Non-optimized parameters: Value in namelist is that used by Snow-17 or SAC for all simulations.

Note, in the process of switching code such that all parameters require three values.

3) RUN CODE

After the namelist is specified following the above as an example, the script *go\_hcdn\_huc02\_seed.bash* can be executed. This script will run all gages listed in the *for loop* for this basin. It takes the base namelist, makes a copy and modifies it accordingly for each basin. Ten random seed starts are run for each basin, resulting in 10 SCE output files that are handled by the script.

Run the script with: *./go\_hcdn\_huc02\_seed.bash*

Parallelization note: I currently have multiple scripts to “parallelize” the code. Each one needs to be started manually. This obviously needs to be improved.

**3) Running for calibration/validation model output**

After all the SCE runs are done for the region: *cd ../*

Current working directory should be: */d2/anewman/run/*

Within this directory is the bash script: *pull\_huc02\_optimal\_simple.bash*

This script will go through all the SCE output for each basin and each seed for user specified (in script) regions. This file is named: *region\_%02d\_opt.txt* where *%02d* is the 2-digit HUC-02 region code. The files are placed in */d2/anewman/hcdn\_output/%02d*

Manual labor is now required to remove gages that had errors in their calibration runs in the *region\_%02d\_opt.txt* file. This is generally due to incomplete streamflow or non-existent forcing data. A few gages fail Kevin's processing scrips due to discrepancies between the GAGES-II set and Roland's PRMS geospatial databases.

Manual labor need to insert the number of total gage/seed SCE runs are left for each region after removal of defunct gauges in the *region\_%02d\_opt.txt* file. This is the number of lines in the file divided by 2.

The calibration/validation model output can now be generated for each gauge/seed pair: *cd 13*

Then run the bash script *go\_huc02\_optimal\_seed.bash* with: *./go\_huc02\_optimal\_seed.bash*

This script needs the *val\_f* variable set to determine whether to run the calibration or validation period (see namelist description). It also will run the Snow-17/SAC model combo only for the seeds and gauges specified in the script. The output file names are created by the executable based off of the *model\_out* namelist parameter. To get calibration and validation model output, this script needs to be run twice, once with *val\_f = 0* and once with *val\_f = 1*

**Generation of metrics & graphics:**

**1) Calculating basin characteristics and model performance metrics**

Potential manual labor: Transfer all the model output and *region\_%02d\_opt.txt* files from hydro-c1 to wherever you run MATLAB. Once you have all the model output where you run MATLAB:

MATLAB script: *calc\_model\_metrics.m* This script will take all the model calibration or validation runs, for all 10 seeds, and compute metrics for the best seed. Metrics like NSE, FDC biases, and also basin characteristics, etc. are all calculated and cataloged and output to a .mat file.

Has a flag for calibration or validation time period and a flag for creating output .mat file.

**2) Station Distance**

a) Grabbed station locations from NCDC for COOP stations and for SNOTEL sites from NRCS. Original data files located on hydro-c1 (and whiteout). *coop\_station\_list.txt, coop\_station\_list\_hist.txt, snotel\_list\_v2.txt*

b) MATLAB script:  *station\_density.m* Will process the COOP and SNOTEL text files and output station lat, lon for each year. Many stations move, stop, start from year to year, so we need yearly station location data.

c) MATLAB script: *station\_count.m* Will take yearly text output from *station\_density.m* and then calculate distances from each basin for each year. Creates .mat files that are used in next phase of metrics & graphics generation.

**3) Initial Graphics**

a) MATLAB script: *plot\_model\_performance\_cal\_val.m* This script generates all the spatial and scatter plots used in my presentations. Uses .mat input from *calc\_model\_metric.m* Has a flag for calibration or validation time period plots.

b) MATLAB script: *cdf\_cal\_val.m*  This script generates all the cdf plots and a couple different scatter plots used in my presentations. Reads in both calibration and validation .mat files from *calc\_model\_metric.m*

**4) Individual Gage Plots**

a) MATLAB script: *precip\_forcing\_compare.m* Will compare Daymet and CPC gauge analysis daily precipitation for a specified gauge and make a few diagnostic time series plots.

Needs as input: GDP lump forcing file and streamflow file used in SCE FORTRAN code. Model output file generated by SCE code. CPC precipitation files and shapefile used in generating GDP forcing file.

b) MATLAB script: *forcing\_streamflow\_qc\_plots.m* Will generate diagnostic time series plots for a specified gauge. Produces water balance and daily squared error time series plots as well. Water balance plots don't include SWE yet...

Needs as input: GDP lump forcing file and streamflow file used in SCE FORTRAN code. Model output file generated by SCE code.

c) MATLAB script: *plot\_basin\_optimal\_streamflow\_simple\_seed.m* Will generate streamflow time series plots for all basins in a region.

Needs as input: GDP lump forcing file and streamflow file used in SCE FORTRAN code. Model output file generated by SCE code. Region SCE optimal parameter output file.