This document is a guide for running the Water Balance code. This code was created by Joseph Crane with coding guidance from Faith Kulzer (WEST) and Hydrology guidance from David Thoma (NPS). It borrows from scripts created by Abigail Volk (NPS), Steve Huysman, Mike Tercek, and John Gross. It modifies the code in the Water Balance package created by Amber Runyon in the Water Balance Functions file. This code combines the Water Balance package with the IHACRES flow function to predict daily stream flow. It runs the IHACRES model on future Water Balance values produced by Mike Tercek. This can be with or without optimizing the Water Balance and IHACRES flow variables. Future Water Balance models are run by Mike Tercek and the optimized IHACRES A and B coefficients are used to get the total daily flow for future years.

In the Water Balance folder, there are a few files and folders. The Code folder has R scripts that are sourced when running the Full Script, Wrapper Script, and Sensitivity Analysis files. The Data folder has data files that are needed for the code to run. Additionally, files are scraped and stored there when running the Full Script or Wrapper Script. The output Folder starts out empty and is filled dynamically with RDS objects and pdfs of plots according to the watershed location and other variables. The RExcelCrossCheckWetBeaver folder houses an R script, an Excel model, and a word document explaining that the calculations in the R code are validated with the Excel Model in the Wet Beaver Creek location. The Water Balance Folder also holds the Full Script and the Wrapper Script. The Full Script has all the code needed to run the Water Balance Model, but the Wrapper Script sources the Wrapper Function which is one large function taking in all the code in the Full Script. Basically, you can run the Full Script for one location, save the results, and run it again. The Wrapper Script allows you to run the Water Balance code at many different locations or with many settings of different variables all at once in a for-loop.

You will have to make sure you have the Water Balance and ClimateR packages installed. Amber Runyon, the maintainer of the Water Balance package recommends using R 4.3.2. Links for their github repositories are below. There are directions on those pages for the installations.

* [GitHub - CCRP-Adaptation/WaterBalance: R package for implementing Dave Thoma's water balance spreadsheet model](https://github.com/CCRP-Adaptation/WaterBalance)
* [GitHub - mikejohnson51/climateR: An R 📦 for getting point and gridded climate data by AOI](https://github.com/mikejohnson51/climateR)

I will talk in depth about the code in the Full Script but only briefly about the code in the Wrapper function because they are so similar. In the Full Script, there are comment headers to orient the user. I have put them all in a bulleted list below. They show the big picture of the code.

* Set path variables and source in files
* Set variables
* Define variables that do not need to be defined outside of the function
  + These rely on other variables that got set in the previous section
* Scrape and format USGS stream gauge data and GridMet or DayMet data
* Aggregate Gauge Discharge data monthly
* Set Updated Initial Flow Conditions if desired
* Optimize Water Balance variables according to the NSE of monthly summed measured versus modeled stream flow
  + The optimized values are saved and used in the next optimization section of code
* Run the model and save the results with initial/non optimized variables for when not optimizing
* Perform the second optimization.
  + The IHACRES A and B coefficients are optimized according to the NSE of Daily total historic stream flow and daily historic modeled stream flow
* Store the results if not optimizing
* Historic flow plots
  + Create plots comparing Measured to Modeled Flow on a monthly time-step
* Plot historical Modeled Flow on a monthly and yearly time-step
* Read in the future flow from Mike Tercek, use the optimized IHACRES flow coefficients to obtain projected stream flow
* Make Daily, Monthly, and Annual xts plots of stream flow
* Save optimization data frames and results as RDS files
  + Results is created and saved when optimizing and when not optimizing

On line 28 of the Full Script, the Set Switch Variables section starts. It defines numerous switches which allow the user to decide how the function should be run. These switches are manually switched before each run if running the Full Script. The non switch variables are defined from line 42-76 of the full Script. Update both the switches and non-switch variables when running the Full Script in a new location. The following is a description of the switches and other variables that need more explanation:

* PETMethod
  + Oudin is currently the only PET Method supported, but the WB() function has a switch statement to allow for other PET methods
* Scrape
  + 1 will scrape USGS Gauge and Climate Data and save them as csv's. 0 will read in said csv's.
* optimization
  + 1 to optimize Water Balance and Drainage Variables and 0 to not optimize
* delayStart
  + 1 to delay the starting year for calibration accuracy assessment, 0 to not delay and use all years
* NonZeroDrainInitCoeff
  + typically set to 0. This can set to 1 to use nonzero initial drainage coefficients that are the average of the modeled historic January Flows after the Cutoff Year. If set to 0, the initial values stay at 0.
* incompleteMonths
  + Typically set to 0 to remove incomplete months from the monthly comparison of modeled vs measured flow, can be set to 1 for Excel R verification
* GridMet
  + Set to 1 to use GridMet climate data and 0 to use DayMet Climate data
* fillLeapDays
  + Only applicable to DayMet data. Set to 1 to fill in missing DayMet Leap Days with previous days' data. Set to 0 to remove leap days from USGS Gauge data
* future
  + Set to 1 to use Mike Tercek’s projected Water Balance data and make future flow plots. Set to 0 to only run model with historic data.
    - Only set this to 1 if Mike’s future flow data file is in the data folder for the specific water shed location
* userSetJTemp
  + Set to 1 for the user to define the Jennings coefficient and 0 for it to be automatically extracted from the Jennings tif based on the latitude and longitude of the watershed location.
* percentRedGrid
  + Set to a value between 0 and 1 for the percentage of the Grid to be searched in the NSE Daily optimization. Must be between 0 and 1, 1 inclusive.
* Plot
  + Set to 1 for plotting and 0 if you don't want to create any plots.
* FolderName
  + This will be the name of the folder created inside the location folder for the run.
* Optional scaling factors for GridMet climate data
  + Coefficients have been determined via comparison using a co-located gridcell and weather station accuracy assessment. The linearregression coefficients are determined from a daily time step for Tmin, Tmax, and precipitation.
  + If no scaling is desired set slopes to 1 and bias to zero.

The Full Script can be run multiple times, and the output will be saved into the correct folders if the user changes the FolderName to something new and the ClimateSiteID variable to a new location if necessary. By changing the FolderName variable, the user creates a new Folder for the results of that run to be saved in. If the FolderName variable is not changed, the results of that run will overwrite the results of the previous run.

This is where the Wrapper Function and Script come in handy. They can be used to create a large data frame of all the settings of the variables needed. For example, each row could have a different location for the ClimateSiteId. Or each row could have a different level of some other Water Balance variable or switch. The wrapper function is run iteratively over the created data frame and the RDS files from the runs are saved into the dynamically created folders. In the Wrapper Script, the user will see the object v which houses the variables. It has one row. Run the following code for an explanation of the Wrapper Script. Clear your environment and run lines 1-54 of the Wrapper Script. This loads the libraries and sources in the code. Then run the following, lines 58-60 of the wrapper script.

***v[2, ] = v[1,] # creates a second row identical to the first row***

***v[2,"FolderName"] = "TestDayMet" # changes FolderName in the second row to TestDayMet***

***v[2,"GridMet"] = 0 # changes the GridMet variable to 0 in the second row***

Next, run lines 62-83 of the Wrapper Script. This iterates over all rows of the v data frame running the model and saving the results each time. It will run once with GridMet Data and once with DayMet data because of lines 58-60.

The Full Script should be used instead of the Wrapper Script when optimizing. The Wrapper Script encounters an error when optimizing. This is because when creating the WBcoeffs object in the 1st optimization routine, the WBcoeffs argument must not be passed so all rows will be saved. Not passing WBcoeffs means that the Wrapper Function can’t find the object WBcoeffs.

Once you have saved the RDS files to their folder directories, to read them in later for sensitivity analysis, you will have to create a location object. outPath will already be defined from the code in the Full Script. Using file.path(), create the full path to where the RDS files are saved and read them in. This exact code is seen in the sensitivity analysis file.

***outLocationPath = file.path(outPath, "LocationName", "FolderName")***

***results = readRDS(file = paste0(outLocationPath, "/results.rds"))***

***WBcoeffs = readRDS(file = paste0(outLocationPath, "/WBcoeffs.rds"))***

***IHcoeffs = readRDS(file = paste0(outLocationPath, "/IHcoeffs.rds"))***