WEPPCLIFF v1.4 Documentation

by Ryan P. McGehee © 2019

Updated: 21 November 2019

Available at: https://github.com/ryanpmcg/WEPPCLIFF

The **WEPP CLI**mate **F**ile **F**ormatter (**WEPPCLIFF**) is an R-based command line tool that was developed in order to assist in the automated generation of WEPP climate input files, which are relatively format intensive. Although this was the original function of WEPPCLIFF, it has now evolved into a much more capable tool for creating climate inputs for both WEPP and USLE-family soil loss models. It is my hope that this tool will encourage greater utilization of soil loss models driven by more recent, modern, and densified climate observations. This file provides a comprehensive documentation of the tool, its capabilities, the installation procedure, tutorials, development notes, and acknowledgements.

This work was supported by USDA ARS at the National Soil Erosion Research Laboratory (NSERL).

Document Contents

OVERVIEW

Primary Purpose and Capabilities of WEPPCLIFF

WEPPCLIFF Primary Purpose

WEPPCLIFF was designed with the main purpose of generating climate input files for WEPP (.cli) from more generic climate inputs. The climate input file for WEPP requires at least (sub-hourly precipitation, daily maximum and minimum temperature, daily solar radiation, and daily dew point temperature). However, other variables of at least daily resolution (wind velocity and wind direction) may also be included in the .cli file. Station metadata is also present in the climate input file, which can impact WEPP model outputs.

An example climate input file is shown below, but one should reference the WEPP User Guide if more detailed information is required. The User Guide is available here: https://www.ars.usda.gov/midwest-area/west-lafayette-in/national-soil-erosion-research/docs/wepp/wepp-model-documentation/

Example WEPP Climate Input File (Continuous)

HEADER

```
 0.00
   1   1   0
Station: Anjeni/Ethiopia
 Latitude Longitude Elevation (m) Obs. Years    Beginning year  Years simulated
   37.31   10.40          2405           14          1986             1
 Observed monthly ave max temperature (C)
 27.77 27.57 27.00 25.50 27.39 22.00 20.65 19.94 22.53 24.61 27.00 27.23
 Observed monthly ave min temperature (C)
  6.81  9.46  9.32 10.70 10.84 11.33 10.68 10.35 10.07  9.03  7.63  5.65
 Observed monthly ave solar radiation (Langleys)
 491.5 502.7 517.3 495.6 475.5 408.7 313.9 326.6 405.2 466.2 451.9 455.9
 Observed monthly ave rainfall (mm)
   0.4   7.2  33.6  10.7  56.5 269.8 336.1 295.1 217.1 108.3  30.5   6.8
   day mon year nbrkpt tmax  tmin    rad   w-vel  w-dir   dew
                (mm)    (C)   (C) (ly/day) m/sec    deg    (C)
```

BODY

```
    1   1  1986   2    26.00    6.00 509.0    1.50  270.0  13.4
17.20     0.000
17.40     0.400
    2   1  1986   0    27.00    7.00 502.0    1.50    0.0  12.2
    3   1  1986   0    27.00    6.00 498.8    2.25   90.0  13.0
    4   1  1986   0    28.00    8.00 528.6    4.00   90.0  12.7
    5   1  1986   0    28.00    8.00 531.6    1.50  225.0  12.5
    6   1  1986   0    27.00    7.00 520.0    4.75  315.0  11.8
    7   1  1986   0    27.00    7.00 476.3    2.25   90.0  13.9
    8   1  1986   0    26.00    6.00 458.6    1.50  270.0  13.7
    9   1  1986   0    27.00    5.00 415.1    0.75   90.0  13.5
   10   1  1986   0    27.00    6.00 454.6    1.50   45.0  13.4
   11   1  1986   0    26.00    7.00 469.0    2.25  202.5  14.2
   12   1  1986   0    28.00    6.00 440.9    1.50   45.0  14.3
   13   1  1986   0    28.00    5.00 451.7    1.50   45.0  13.5
   14   1  1986   0    27.00    5.00 505.8    1.50   45.0  13.7
   15   1  1986   0    29.00    9.00 521.6    1.50   90.0  13.5
```

Figure 1. *An example climate input file (.cli) for WEPP. There are a few notable variations in the structure of .cli files (most importantly continuous vs. single event and breakpoint vs. non-breakpoint formats). Despite these variations, all .cli files contain a header (where more general information is stored) and a body (which contains the more pertinent daily and sub-daily data used in WEPP calculations).*

General Capabilities

Although the primary purpose for WEPPCLIFF is to process general climate inputs into a specific format (.cli), other capabilities were added to WEPPCLIFF as they were deemed relevant to the primary purpose or that would aid in interpreting the data. A comprehensive list of all accepted arguments, syntax, and methods is included later, but a broad summary of these capabilities is listed below:

Basic Capabilities (Always Performed at Some Level)

- Accepts any delimited text input
- Accepts any precipitation input format
- Accepts most common metric and English units
- Accepts any
- Storm separation can be performed for any time and depth threshold
- Parallel processing
- Climate file customization

Optional Capabilities (Performed Only When Invoked)

- Quality Checking
- Gap Detection
- Gap Filling
- Visualization
- Export to File
- Erosion Indices
- Energy Equation

Important Distinction

WEPPCLIFF was developed in part to replace a pair of niche tools called Breakpoint Climate Data Generators (BPCDG) and (BPCDG2). Although the naming of these tools may lead to some confusion, **neither BPCDG nor BPCDG2 actually *generate* 'breakpoint climate data'**. **What these tools can do is *reformat* observed climate data into a 'breakpoint data format'**. This is still a very useful service to a user, but perhaps less useful than a tool that can actually generate breakpoint climate data, such as that from a pluviograph. **WEPPCLIFF (1.0 and later) can actually generate stochastic breakpoint climate data when sufficient observed data are provided and the multiple imputation gap filling model is invoked by the -id flag.**

The phrase 'breakpoint data' arises from periods of constant intensity in precipitation data, where it is important to know how intensity changes over time. The more common alternative is 'fixed-interval' data, which average changes in intensity over a period of time. Therefore, a higher resolution input will result in closer approximations of breakpoint data, but this should not be confused with nor is it worthy of the title 'breakpoint data'. This is the reason for the name change to WEPPCLIFF (**WEPP CLI**mate **F**ile **F**ormatter), as well as to more closely associate the tool and its purpose with WEPP. WEPPCLIFF can do much more than merely format a climate file, but those functions really only enhance its ability to deliver on the primary purpose, which is to create climate inputs from observed data for use in soil loss models.

INSTALLATION

Running WEPPCLIFF for the First Time

WEPPCLIFF Installation and Verification Procedure

There are a few steps that need to be completed before you are ready to run WEPPCLIFF. Make sure you have the necessary files from the WEPPCLIFF GitHub: https://github.com/ryanpmcg/WEPPCLIFF. These files include:

1. WEPPCLIFF.pdf – Documentation
2. WEPPCLIFF.zip – Tutorial and Code

Once you have the files above, *unzip WEPPCLIFF.zip to the C:\ drive on your machine. If you are using another path or OS, you must keep the path to this folder for later.* A tutorial follows, which assumes the folder was unzipped to the C:\ drive on a Windows machine. If you modify any step, you will be responsible for knowing how to proceed. Non-web-based installation alternatives are discussed at the end of this section.

FOLLOW INSTRUCTIONS IN RED FOR A SUCCESSFUL INSTALLATION AND VERIFICATION!

Installation / Verification Outline

1. Install R source code (also called 'base R')
2. Run WEPPCLIFF tutorial (Tutorial.txt) in your terminal which includes
    - Installing dependencies for the first run
    - Running many examples (quickly)

Installing R

WEPPCLIFF.R (v.0.1) was built on R (v.3.3.3). WEPPCLIFF will be intermittently updated to support newer versions of R as its underlying dependencies support newer versions (WEPPCLIFF v1.0+ was updated to R v.3.6.1. It is easy to 'rollback' to older versions of R or conversely to update to newer versions if necessary. You can search for how to do this in almost any forum. If you would like to build a system from scratch (recommended for most users) follow these instructions:

Pick a CRAN mirror near you here: https://cran.r-project.org/mirrors.html then install R-3.6.1 (recommended) or a newer version if you prefer, you can find OS-specific R installation files in your selected CRAN mirror. Use a typical installation. You can verify that installation was successful by opening R on either system and a user terminal will open with version and licensing information about R. Please check now, and verify that you can run R. If you cannot, you will not be able to complete installation.

Using the Automated Tutorial (Not Recommended; You Won't Learn As Much!)

Now that R is installed on your system, your system can now understand most of the WEPPCLIFF code, but there are some important components missing. If you want to skip the tutorial and are using a Windows machine, you can just paste the contents of ***AutoTutorial.txt*** in your terminal or double-click ***SkipTutorial.bat*** to run the commands without interaction. A ***AutoTutorial.sh*** file is provided for UNIX systems which only removes the 'setx' command from the Windows equivalent.

Running the WEPPCLIFF Tutorial as Intended

Open ***Tutorial.txt*** and copy each command (one-by-one) into your terminal. Read the comments in the tutorial for a brief explanation and reference this document for a more thorough explanation.

WEPPCLIFF is a command line tool, and in order to allow user provided arguments to be passed into the WEPPCLIFF.R script (you will learn about these later), you need to use Rscript. If you are using Windows, you may want to add the path to Rscript to your 'system path' for convenience when calling Rscript (if you are using OSX, you can skip this step). In OSX, Rscript is stored in a way that it is searched by default and does not have to be explicitly provided for OSX. The following command adds the default path to Rscript to the system variable 'PATH':

```
setx PATH "%PATH%;C:\Program Files\R\R-3.6.1\bin\"
```

Now close and restart your terminal. If the path was successfully added, you will be able to type 'Rscript' into your terminal and the tool options will print to screen (regardless of your working directory). If the path was added only temporarily or if it was not added, you will need to type the full path every time you use Rscript. This is the default path to Rscript for Windows (check your version of R):

```
"C:\Program Files\R\R-3.6.1\bin\Rscript.exe"
```

HINT: After executing any command in 'command prompt' or 'terminal' you can press the 'up arrow' key to retrieve your last command, then you can add to or modify that command to save time.

The following command changes to the root WEPPCLIFF directory (should be where you unzipped the WEPPCLIFF.zip folder at the beginning of the installation procedure).

```
cd C:\WEPPCLIFF
```

Now you are ready to interact with WEPPCLIFF code for the first time. The following command uses Rscript to engage WEPPCLIFF.R (the WEPPCLIFF source code) and passes the flag ('-fr' for FIRST RUN) and matching argument ('t' for TRUE). This tells WEPPCLIFF that it needs to install its dependencies.

```
Rscript --vanilla WEPPCLIFF.R --args -fr t
```

The installation time will vary based on your system hardware and internet connection. The process can take more than 10 minutes (for slow internet connections or older hardware) and as little as 30 seconds (for faster connections or newer hardware). Windows permissions sometimes prevents this step from working seamlessly, which is why the folder needs to be installed on the C:\ drive directly. Also, depending on the version of R that you are using, some packages may not have had time to update to the new version or no longer support the new version of R (this is rare). In that case, you can always roll back your R version, delete the C:\WEPPCLIFF folder, and reinstall.

Installation Alternatives

An internet connection is recommended for ease of installation, but in cases where there is no web connection, the dependencies can be installed manually. You can easily find information about how to do this for non-networked machines. After installation is complete, you will not need a web connection and packages will remain on your system until removed.

EXAMPLES

Exploring WEPPCLIFF Capabilities

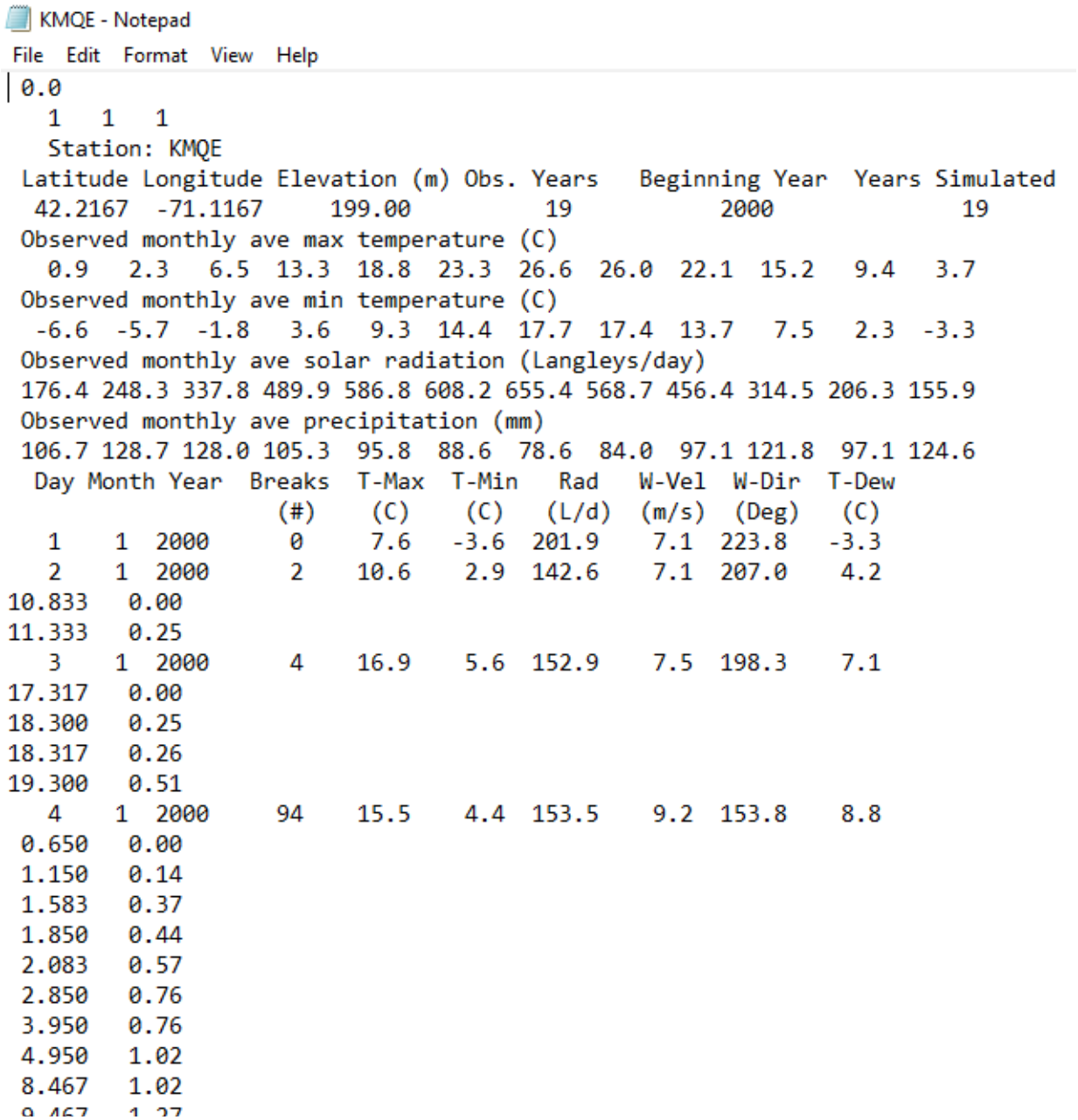
Basic Functions

The following examples will introduce you to some of the basic functions of WEPPCLIFF. For example, this portion of the tutorial will show you how to control the input and output file names and directories as well as reading various datetime formats and analyzing different periods within your input file. As always, do not copy from this document to your terminal since some characters can be changed during the copy and paste. These are only included so you can reference the Tutorial.txt in more detail.

Run the following input file with default settings. This should take about 2 minutes.

```
Rscript --vanilla WEPPCLIFF.R --args -f ASOS_BPT_KMQE.csv
```

If you were to open the output “KMQE.cli” file (in the output folder) you should see this output:



Now we will run the same example again but with a different output name and we will provide early license agreement (so that the license agreement prompt will not be triggered). Let's also turn verbosity on so that we can watch what is happening more closely.

```
Rscript --vanilla WEPPCLIFF.R --args -f ASOS_BPT_KMQE.csv -fn bpt -la y
-verb t
```

You should notice that most of the time is spent writing the .cli file. This is one of the most time-consuming portions of WEPPCLIFF, and the input we are using is real breakpoint data (much more information than most fixed interval data). This saves all WEPPCLIFF output by the name you provide.

Now let's do that again, but we will save the output to a different directory, turn on graphical output, and specify a shorter time period for analysis.

```
Rscript --vanilla WEPPCLIFF.R --args -f ASOS_BPT_KMQE.csv -la y -verb t
-o C:/WEPPCLIFF/JUNK -pd t -sdt "2000-01-01 00:00:00" -edt "2004-12-31
24:00:00"
```

This file (ASOS_BPT_KMQE.csv) uses the same datetime format as the WEPPCLIFF default (%Y-%m-%d %H:%M:%S; that is four digit year, hyphen, one or two digit month, hyphen one or two digit day, space, one or two digit hour, colon, one or two digit minute, colon, and one or two digit second); therefore we do not need to provide datetime format flags and arguments (i.e. -dtf1, -dtf2, or -dtf3 and format strings). When we are providing a start and end date, we must use the same format as -dtf1 (listed above). Let's only analyze a five-year period from 1 January 2000 to 31 December 2004.

You can open the input file in a text editor to view the format to verify that it is the same. If you use other programs, such as Microsoft Excel, the text may not be displayed correctly due to conventions used in those programs. Also, you should never save after opening the file in such a program because it can change the format of the data, which would then need a different format string for WEPPCLIFF to read it correctly. The lesson for today: USE A TEXT EDITOR!

This concludes the basic functionality tutorial. Feel free to try other commands to explore WEPPCLIFF capabilities for yourself. The next two sections will look at two different types of precipitation data, which is the most important climate driver of soil loss by water.

Breakpoint Inputs

This section of the tutorial will show you how to process breakpoint precipitation data in WEPPCLIFF. This document does provide an exhaustive discussion of precipitation data. However, in demonstrating WEPPCLIFF capabilities, it is necessary that you have some understanding.

Breakpoint data (as opposed to "breakpoint-formatted" data) refers to precipitation data where periods of constant characteristics are recorded between 'breaks'. For precipitation data, the constant characteristic is almost always intensity, but it could, in theory, be some other characteristic.

Breakpoint-formatted data refers to data that has been formatted exactly the same as breakpoint data, but there is no guarantee of constant characteristics between breaks. That is essentially what fixed-interval data is: breaks are inserted at regular intervals with no regard for what characteristics were actually observed.

There is a somewhat famous example (in soil loss modeling literature) of breakpoint precipitation data in an older USDA ARS publication (Wischmeier and Smith, 1978; known casually as Agriculture Handbook No. 537 or AH537). This kind of data is difficult to obtain at scale, and there are much fewer recording stations for breakpoint data as opposed to fixed-interval data. We will be using the example storm from AH537 for the next two sections of the tutorial, which has been carefully formatted as an appropriate input for WEPPCLIFF.

The first thing you should know is that WEPPCLIFF assumes breakpoint inputs. The precipitation interval flag (-pi) defaults to 'f' for false, meaning there are no fixed intervals. Run the following example file which used cumulative precipitation inputs, and mixed units within the file columns (metric inputs of daily data and English inputs for precipitation).

```
Rscript --vanilla WEPPCLIFF.R --args -f AH537_BPT_CP.csv -cp t -u m
PRECIP -sm 2 -la y
```

Now let's rerun the same example, but let's turn on graphical output, erosion index calculations (using the AH537 energy equation), and export functionality.

```
Rscript --vanilla WEPPCLIFF.R --args -f AH537_BPT_CP.csv -cp t -u m
PRECIP -sm 2 -la y -verb t -ei t -ee AH537 -pd t -ed 3
```

Check out the graphical output under C:/WEPPCLIFF/PLOTS/OTHER/ and look at the exported precipitation data under C:/WEPPCLIFF/EXPORT/ before moving on.

Now, let's run the same data but let's use the incremental precipitation formatted file and flag.

```
Rscript --vanilla WEPPCLIFF.R --args -f AH537_BPT_IP.csv -cp f -u m
PRECIP -sm 2 -la y -verb t -ei t -ee AH537 -pd t -ed 3
```

You should notice that the outputs and plots are exactly the same, with perhaps the exception of rounding error, which is very small for the vast majority of WEPPCLIFF applications.

This concludes the section for utilizing breakpoint precipitation data. The most important thing to note about breakpoint data (from the plots of intensity vs time) is that the intensity is constant (flat line) between breaks AND that those are actually representative of the precipitation that occurred between the breaks. Now let's look at this same storm represented by fixed-interval inputs to WEPPCLIFF.

Fixed Interval Inputs

In this section you will run several variations of the same command as before but using various fixed interval products instead of breakpoint inputs.

The following command will use a **one-minute fixed-interval dataset that has unlimited depth precision** (this type of gage does not actually exist, but it is included to illustrate a point about time and depth limitations of fixed-interval data). Run the following command.

```
Rscript --vanilla WEPPCLIFF.R --args -f AH537_1MIN_UP.csv -pi 1 -cp f -
u m PRECIP -sm 2 -la y -verb t -ei t -ee AH537 -pd t -ed 3
```

This command is also **one-minute data** but it has the **modern depth precision of 0.01 inches or 0.254 mm, which was widely implemented in the US beginning in November 1993**. Accuracy of these gages is also important, but it is not discussed here for brevity. Run the following command.

```
Rscript --vanilla WEPPCLIFF.R --args -f AH537_1MIN.csv -pi 1 -cp f -u m
PRECIP -sm 2 -la y -verb t -ei t -ee AH537 -pd t -ed 3
```

The next command uses a **five-minute interval with the same precision of 0.01 inches**.

```
Rscript --vanilla WEPPCLIFF.R --args -f AH537_5MIN.csv -pi 5 -cp f -u m
PRECIP -sm 2 -la y -verb t -ei t -ee AH537 -pd t -ed 3
```

The next command uses a **ten-minute interval with the same precision of 0.01 inches**.

```
Rscript --vanilla WEPPCLIFF.R --args -f AH537_10MIN.csv -pi 10 -cp f -u
m PRECIP -sm 2 -la y -verb t -ei t -ee AH537 -pd t -ed 3
```

The next command uses a **fifteen-minute interval with the same precision of 0.01 inches**.

```
Rscript --vanilla WEPPCLIFF.R --args -f AH537_15MIN.csv -pi 15 -cp f -u
m PRECIP -sm 2 -la y -verb t -ei t -ee AH537 -pd t -ed 3
```

The next command uses a **fifteen-minute interval with unlimited precision**.

```
Rscript --vanilla WEPPCLIFF.R --args -f AH537_15MIN_UP.csv -pi 15 -cp f
-u m PRECIP -sm 2 -la y -verb t -ei t -ee AH537 -pd t -ed 3
```

The last command uses a **fifteen-minute interval with the pre-modern precision of 0.1 inches**.

```
Rscript --vanilla WEPPCLIFF.R --args -f AH537_15MIN_HT.csv -pi 15 -cp f
-u m PRECIP -sm 2 -la y -verb t -ei t -ee AH537 -pd t -ed 3
```

Now go through and look at each of the graphs produced by WEPPCLIFF and compare the results. Pay close attention to the timing and magnitude of intensity from each dataset. This was a very large storm with 1.3 inches of rain falling in 90 minutes (averaging 0.867 inches per hour with peak rainfall of 3.00 inches per hour). The highest recorded and confirmed rainfall in the US is 1.27 inches of rain in one minute or 76.2 inches per hour. The 15-minute data peaked at 2.4 inches per hour and the timing was different. How do you think that will impact the hydrology calculations? Soil loss? Chemical transport? Etcetera, etcetera?

Since this was a large storm it does not really show the impact of depth precision on smaller storms, which tend to be lower intensity. Did you know that precipitation energy is a logarithmic function of intensity (most of the change in energy occurs in low intensities)? How do you think the fixed-interval data with coarse depth precision will perform with smaller events?

Can you guess which one of all the above inputs is used for national soil conservation planning efforts in the US (as of 15 November 2019)...

So, the next time you are tempted to think that advances in soil loss modeling (or any of the physical sciences for that matter) has plateaued, at least remember this as one of the big reasons for that. Now, I hope you understand why tools such as WEPPCLIFF are sorely needed.

Advanced Functions

The last section of this tutorial is really just a cherry on top of the basic WEPPCLIFF functionality, but it is an excellent addition that is also extremely useful. This section will show you how to perform basic quality checking, gap filling, and how to export data for more generic purposes.

Let's start by running the same station we did earlier, but lets run it with quality checking and graphical output on.

```
Rscript --vanilla WEPPCLIFF.R --args -f ASOS_BPT_KMQE.csv -la y -verb t
-qc t -pd t
```

Now check the input plots (C:/WEPPCLIFF/PLOTS/INPUT) and you will see the portions that have passed quality checking are in red while the original data are drawn in black. You may notice some quality problems still exist. As of version 1.4, WEPPCLIFF only supported limited quality checking. Several other routines from previous versions were commented out until they could be refactored and integrated with the new code structure in version 1.4.

Now, run the example again with gap filling (quick version) and binary data export.

```
Rscript --vanilla WEPPCLIFF.R --args -f ASOS_BPT_KMQE.csv -la y -verb t
-qc t -pd t -id t -qi t -ed 1
```

You can see that the runtime increases significantly, but the total runtime for a single station (executed in series) is still only about 3 minutes (plus or minus for your particular machine build). You can also check the output plots (C:/WEPPCLIFF/PLOTS/OUTPUT) and you will see the portions that were original in black and those which were filled in red.

Lastly, if you are very patient, run the example again with gap filling, but we will use the super-tedious, has-to-be-right, slightly-OCD version with filling verbosity on (so you won't think it is frozen ☺).

```
Rscript --vanilla WEPPCLIFF.R --args -f ASOS_BPT_KMQE.csv -la y -verb t
-qc t -pd t -id t -iv t -ed 1
```

This is using the recommended settings from statistical literature for iterations based on the ratio of data observed to missing and it is using imputation methods that I found to be marginally better for filling data but at sometimes extreme costs to computation time. Generally speaking, even runtimes for 1-minute inputs of all variables, extreme gaps, and a total observation period of 100-years, should complete in less than a few hours. You can decide how precious those marginal improvements are and whether or not they are worth the computation time.

Now you have completed the WEPPCLIFF tutorial. Please remember to read the rest of this document for more information about other capabilities, inputs, outputs, etc. If you would like to try to open one of the binary export files, please follow the following instructions. Accessing other export options are not supported in this tutorial.

Begin an R session in either RStudio or your terminal (whichever you prefer) and load the binary data using the following:

```
data = readRDS("C:/WEPPCLIFF/EXPORT/KMQE.rds")
```

Subset data to obtain elements from the primary list structure using some variation of the following command (substitute a name of your choice for 'element' and a number 1-22 for 'x'):

```
element = data[[x]]
```

Thank you for completing the tutorial. Please direct any comments, criticism, or questions to the Github page.

FUNDAMENTALS

About Arguments and Syntax

Command Line Tools

Experienced programmers: the WEPPCLIFF tool functions much like any other shell script tool; it accepts arguments and executes various processes based on those arguments.

Beginners: read this section to understand the basics of a command line tool.

In the WEPPCLIFF tool an 'argument' consists of 1) a flag (to denote one or more arguments that follow) and 2) the argument itself. If not specified correctly, some flags will cause the script to fail; others may alter the output erroneously. A complete table of arguments is provided below.

Syntax:

All arguments must be passed into a terminal or command prompt in the following format (description of each item is provided below):

```
Rscript --vanilla WEPPCLIFF.R --args <WEPPCLIFF ARGS>
```

Rscript	A wrapper function that calls an R script to run via the command line
--vanilla	An argument to Rscript, sets up R environment for running scripts
WEPPCLIFF.R	The path to the actual script we want to run
--args	A flag for Rscript, telling it that arguments to WEPPCLIFF.R follow
WEPPCLIFF ARGS	Must be entered as follows: "-$flag_1$ arg_1 -$flag_2$ arg_2…-$flag_n$ arg_n"

General Notes:

Here is a brief list of important syntax notes for building WEPPCLIFF commands:

- flags must be signaled with a preceding hyphen
- flags must be paired with their arguments (sequentially)
- flag / argument pairs must be separated by one or more spaces
- flag / argument pairs can be provided in any order
- unspecified flag / argument pairs always have a default (detailed on following pages)
- arguments that contain spaces within them must be double quoted (e.g. "arg with spaces")

Common Mistakes:

Here is a brief list of the most common mistakes to avoid in WEPPCLIFF commands:

- misspelled flags or arguments
- commands copied from non-plain text sources (e.g. Word, a .pdf, etc.)
- path to Rscript.exe and/or WEPPCLIFF.R not included or not stored in PATH
- operating in a directory that has restricted permissions
- updating version of R without reinstalling WEPPCLIFF dependencies
- inconsistent datetime format strings and input datetime data
- not reading the documentation
- not checking this section carefully
- trying to cut corners in the learning process

Table of Arguments (Continued on Next Pages; Last Updated in WEPPCLIFF v1.4):

INSTALLATION ARGUMENTS

FLAG	NAME	DEFAULT	OPTIONS	DESCRIPTION
fr	first run	**F**	T / F	determines whether or not dependencies should be installed; this should be T for the first run and F for all subsequent runs on a machine
la	license agreement		Y / N	the default is to not provide a response to the license agreement

INPUT / OUTPUT ARGUMENTS

FLAG	NAME	DEFAULT	OPTIONS	DESCRIPTION
d	input directory	**./INPUT**	a directory	the parent directory of the single input file or the directory of files to process; the WEPPCLIFF home directory is the default local root
o	output directory	**./OUTPUT**	a directory	the location where all .cli files will be written; the WEPPCLIFF home directory is the default local root
e	export directory	**./EXPORT**	a directory	the location where all binary .rds (binary) or .json (text) files will be written; the WEPPCLIFF home directory is the default local root
p	plot directory	**./PLOTS**	a directory	the location where all graphical output files will be written; the WEPPCLIFF home directory is the default local root
f	filename	**NONE**	.txt, .csv, or .tsv file	the input file if there is only one; leaving this blank will cause the program to try and process all files in the input directory; accepts tab or comma separated entries (must be .tsv or .csv, respectively) or you can specify a delimiter for .txt files with -delim
fn	output filename	**station**	a string	the name which will be used as default for all files written during WEPPCLIFF processing; specifying 'station' will search the input data for a station name to be assigned to the outputs (especially useful for parallel processing)
delim	file delimiter	**a single space**	a string	specifies a delimiter to use when the input file is of type .txt

FUNCTIONALITY ARGUMENTS

FLAG	NAME	DEFAULT	OPTIONS	DESCRIPTION
u	units	**M**	M <vars> / E <vars>	the units of the input file (Metric or English) followed by variables which are exceptions must be specified in order for the correct conversions to be applied
qc	quality control	**F**	T / F	determines whether or not to perform quality control routines for the input data; the flags -qcop, -qcth, and -qcdf control the methodology
id	impute missing data	**F**	T / F	determines whether or not to engage a multiple imputation model to fill missing data; the flags -im, -io, -qi, and -iv control the methodology
pd	plot data	**F**	T / F	determines whether or not to create graphical output for both inputs to WEPPCLIFF and resulting output data
ed	export data	**0**	0 / 1 / 2 / 3 / 4 / 5	determines whether or not to generate binary or human readible, output of internal calculations; (0) no files written; (1) .rds binary files written; (2) .json text files written; (3-5) .csv text files written with precipitation, daily, or storm timeseries dataframes, respectively
alt	alternative data	**F**	T / F	specifies whether or not to use alternative data to calculate minimum, maximum, and dew point temperatures; this requires the variables AIR_TEMP and REL_HUM
pmd	preserve missing data	**F**	T / F	specifies whether or not to maintain NA values throughout execution; when true, this converts all non-numeric inputs and calculations and the WEPPCLIFF missing data value (-99999) values to NA; internal calculations and plots are affected according to various conventions throughout execution

PRECIPITATION ARGUMENTS

FLAG	NAME	DEFAULT	OPTIONS	DESCRIPTION
cp	cumulative precipitation	**F**	T / F	specifies whether or not input precipitation data is cumulative (T) or incremental (F)
pi	precipitation interval	**F**	F / positive integer	specifies a precipitation time interval in minutes to use; (F) indicates that the data provided are already in breakpoint format (i.e. the interval is exactly the difference between measured values
ei	erosion index	**F**	T / F	specifies whether or not erosion indices should be calculated
ee	energy equation	**R2**	ALL / ARS / MM / BF / AH282 / AH537 / AH703 / R2 / USER <exp>	specifies the energy equation to be used to calculate erosion indices; options include: Brown-Foster (BF), McGregor-Mutchler (MM), any of the three Agricultural Handbooks (282/537/703), the RUSLE2 version of the BF equation (R2) or a user specified equation (which must be followed by an acceptable Base R expression in terms of intensity as 'i'); an option for all equations (ALL) including the USER equation is available (USER is NULL if unspecified) and (ARS) uses all equations except for the USER equation.
eb	export breakpoints	**F**	T / F	specifies whether or not to save precipitation breakpoints as a .csv file in the data export directory

STORM CONTROL ARGUMENTS

FLAG	NAME	DEFAULT	OPTIONS	DESCRIPTION
sid	storm identifier	**1**	1:n	specifies which storm to use when -sm 2 (single event) is specified; this changes according to the storm separation method and parameters; this can also be used with -sdt for a known start time
tth	time threshold	**6**	a positive number	the time threshold (in hours) to be used for summing precipitation in storm separation
dth	depth threshold	**1.27**	a positive number	the depth threshold (in mm) to be used for determining breaks between storms

QUALITY CONTROL ARGUMENTS

FLAG	NAME	DEFAULT	OPTIONS	DESCRIPTION
qcop	quality checking option	**B**	B / P / E	specifies the quality control method to use where values exceeding critical limits are called outliers and are removed; P allows physical rate and range limits to be applied to each of the 9 variables; E initiates several empirical tests to apply including: spiking, streaking, sticking, and return period (some additional control is possible through the other quality control arguments); B, the default, applies both
chkth	checking threshold	**0.9**	a positive number	specifies the fraction of rate data (lowest %) that can be ignored during quality checking for spiking
spkth	spiking threshold	**0.9**	a positive number	specifies the fraction a potential 'spike' must recover in order to be treated and removed as an actual 'spike'; not applied to precipitation
strth	streaking threshold	**2**	a positive integer	specifies the minimum number of consecutive identical values to be treated and removed as a 'streak'; not applied to precipitation
stkth	sticking threshold	**4**	a positive integer	specifies the number of non-consecutive values over which to check for a potential 'stick'; not applied to precipitation
rp	return period	**1000**	a positive number	specifies the return period in years above which all values should be removed from the dataset as 'outliers' or 'unrepresentative' values; not yet released; only applied to precipitation

IMPUTE CONTROL ARGUMENTS

FLAG	NAME	DEFAULT	OPTIONS	DESCRIPTION
im	impute method	**DEFAULT**	see 'mice' help	specifies the method to use for those imputation models which are not restricted by WEPPCLIFF; the default is to use the original imputation model design for WEPPCLIFF
io	iteration override	**100**	a positive integer	a maximum number of iterations to use for imputation; WEPPCLIFF will never use less than 10 iterations ; the default number of iterations used is determined by the ratio of present data to missing data
qi	quick impute	**F**	T / F	determines whether or not to set parameters for the fastest possible imputation model that is minimally sufficient for reliable results
iv	impute verbosity	**F**	T / F	determines whether or not to print imputation progress; this can be very helpful if there are significant gaps or the data are being used to extend output beyond the observed time period

DATETIME CONTROL ARGUMENTS

FLAG	NAME	DEFAULT	OPTIONS	DESCRIPTION
sdt	start datetime	**first datetime**	a datetime	specifies the start time to which data will be trimmed; if -sm is 1 the data will be trimmed to the first calendar year >= -sdt; if -sm is 2 the data will be trimmed to -sdt exactly
edt	end datetime	**last datetime**	a datetime	specifies the end time to which data will be trimmed; if -sm is 1 the data will be trimmed to the first calendar year <= -edt; if -sm is 2 the data will be trimmed to -edt exactly
rtb	round time bounds	**M**	M / D / H / F	rounds the start and end of complete data to the beginning and ending of the first and last month (M), day (D), hour (H), or does not round (F); this exists to make sure that virtually complete years are not wasted
ipb	ignore precipitation bounds	**F**	T / F	specifies whether or not to include the precipitation timeseries in determining where the data should be trimmed to
dtf1	datetime format 1	**%Y-%m-%d %H:%M:%S**	an R datetime format string	the datetime (or date) format used to process precipitation data; see R Help for more information on formatting options; the default is in the format YYYY-MM-DD HH:MM:SS
dtf2	datetime format 2	**dtf1**	an R datetime format string	the datetime (or date) format used to process alternative data; see R Help for more information on formatting options; the default is the same as that of precipitation data
dtf3	datetime format 3	**dtf1**	an R datetime format string	the datetime (or date) format used to process daily data; see R Help for more information on formatting options; the default is the same as that of precipitation data

CLIGEN FILE FORMAT ARGUMENTS

FLAG	NAME	DEFAULT	OPTIONS	DESCRIPTION
cv	CLIGEN version	**0.0**	CLIGEN version	generally this should be 0.0 because observed data is not from CLIGEN, but the option exists to change this for some WEPP applications
sm	simulation mode	**1**	1 / 2	determines whether WEPPCLIFF should create a continuous CLI file (1) or an event-wise CLI file (2); this can be for a single storm or a series of storms, but this mode has not yet been validated with WEPP
bf	breakpoint format	**1**	1	determines whether WEPPCLIFF output should be written identical to CLIGEN output (0) or in breakpoint format (1); the (0) option is currently unsupported due to the need to verify identical tp and ip calculations; it is possible this will be supported in the future
wi	wind information	**1**	0 / 1	determines whether wind information should be used (1) in the CLI file or ignored (0)

OPTIMIZATION ARGUMENTS

FLAG	NAME	DEFAULT	OPTIONS	DESCRIPTION
mp	multiprocessing	**T**	T / F / # of cores	determines whether or not stations (when there are more than 1) will be processed in parallel; it is automatically disabled for 1 station since not much time would be saved; cores will be determined automatically if not specified directly

DEVELOPER ARGUMENTS

FLAG	NAME	DEFAULT	OPTIONS	DESCRIPTION
prof	profile code	**F**	T / F	specifies whether or not to initiate the profiling routine and to generate profiles in the profiling directory
pint	profile interval	**0.02**	a positive number	specifies the time interval to use when profiling code
mepr	profile memory	**F**	T / F	specifies whether or not to include memory profiling
gcpr	profile garbage collection	**F**	T / F	specifies whether or not to include garbage collection profiling
lnpr	profile lines	**F**	T / F	specifies whether or not to include line profiling
warn	show warnings	**F**	T / F	specifies whether or not warnings should be printed
verb	verbosity	**F**	T / F	specifies the amount of terminal output (essentially provides more execution progress reports to the terminal for single station operation)

INPUTS

Supported Data Input Formats

Acceptable Inputs

WEPPCLIFF is a bit more flexible than the norm regarding input file formats. There are only a few rules which must be obeyed for the input to be accepted. In this section, inputs for single station, multiple station, and parallel processing will be discussed as well as general formatting conventions, which are consistent among all input and processing modes of operation. Examples and summary tables are included to help present information in a concise and ordered fashion. **In short, (and in programming terminology), WEPPCLIFF supports variable width, named, columnar, time series inputs with a few exceptions for metadata inputs, which are scalar or string inputs, in single or multi-file units.**

Supported File Types

WEPPCLIFF supports the processing of tab delimited (.tsv), comma delimited (.csv), or other less common delimited (.txt) data. When providing (.txt) inputs, the delimiter must be specified via [-delim]. Whether processing a single file or a directory, these delimiters (and extensions) are supported.

Supported Formats

Once the file is read into WEPPCLIFF, a generic 'named' format is anticipated. The input convention used in WEPPCLIFF is flexible as compared to input file formats used in similar models or tools. The most important requirement is the variable name, which is used to identify inputs in the input file. Inputs can be unordered (columns randomized), unsorted (rows randomized), contain gaps (can be filled), contain flags (flags will be dropped), contain outliers and errors (can be quality checked), contain various units (as long as each unit is supported), contain different periods of record (only complete records will be used; can be filled to specified start and end times), and some additional features shown above.

Processing Units (Datetime Groups)

Variables are assigned to datetime groups according to how they will be processed by WEPPCLIFF. Precipitation (PRECIP) is always assigned to datetime group (DT_1). Air temperature (AIR_TEMP) and relative humidity (REL_HUM) are always assigned to (DT_2), and all other supported variables (MAX_TEMP, MIN_TEMP, SO_RAD, W_VEL, W_DIR, and DP_TEMP) are assigned to the last group (DT_3).

The following table summarizes the supported formats by datetime group and variable.

FILE FORMAT CONVENTION FOR WEPPCLIFF

PROCESSING UNIT	DT GROUP 1		DT GROUP 2			DT GROUP 3						
VARIABLE NAME	DT_1	PRECIP	DT_2	AIR_TEMP	REL_HUM	DT_3	MAX_TEMP	MIN_TEMP	SO_RAD	W_VEL	W_DIR	DP_TEMP
English Units	NA	in	NA	F	%	NA	F	F	lang/day	mph	degrees	F
Metric Units	NA	mm	NA	C	%	NA	C	C	W/m^2	m/s	degrees	C
Cumulative	NA	YES	NA	NO	NO	NA	NO	NO	NO	NO	NO	NO
Incremental	NA	YES	NA	YES	YES	NA	YES	YES	YES	YES	YES	YES
Fixed Interval	NA	YES	NA	YES	YES	NA	YES	YES	YES	YES	YES	YES
Breakpoint	NA	YES	NA	YES	YES	NA	YES	YES	YES	YES	YES	YES
Minimum Supported Interval (Minutes)	> 0		> 0			> 0						
Maximum Useable Interval (Minutes)	<< 1440		< 1440			1440						
Suggested Interval (Minutes)	< 30		60			1440						

Single vs. Multiple Station Processing

There are a few different ways that one may provide input to WEPPCLIFF for processing. The simplest input is that of a single station which will always be contained within a single file. See the following figure for an example of a single station input file. You will see that this file follows the file format convention shown above. The most important features yet to be discussed are the station metadata.

Following the 'named' file format convention, the station name (STATION), latitude (LAT), longitude (LON), and elevation (ELEV), must be provided under the correct name. Data beyond the first observation (the first row beneath the header) will be ignored in a single station scenario. Elevation can be specified in English or metric units (ft or m), respectively. Therefore, it should also be specified in the [-u] argument for unit conversion (if required).

DT_1	PRECIP	DT_2	AIR_TEMP	REL_HUM	DT_3	W_DIR	W_VEL	SO_RAD	STATION	LAT	LON	ELEV
1/1/06 0:00	0	1/1/06 0:00	33.62	93	1/1/06 0:00	226	4.1	0	AS1	41.5	-82.5	200
1/1/06 0:10	0	1/1/06 0:10	33.62	94	1/1/06 0:10	252	4	0				
1/1/06 0:20	0	1/1/06 0:20	33.58	94	1/1/06 0:20	225	5.7	0				
1/1/06 0:30	0	1/1/06 0:30	33.42	95	1/1/06 0:30	212	4.7	0				
1/1/06 0:40	0	1/1/06 0:40	33.22	95	1/1/06 0:40	227	5.6	0				
1/1/06 0:50	0	1/1/06 0:50	33.38	95	1/1/06 0:50	234	6	0				
1/1/06 1:00	0	1/1/06 1:00	33.46	94	1/1/06 1:00	220	6.3	0				
1/1/06 1:10	0	1/1/06 1:10	33.38	93	1/1/06 1:10	250	6.8	0				
1/1/06 1:20	0	1/1/06 1:20	33.38	93	1/1/06 1:20	250	6.6	0				
1/1/06 1:30	0	1/1/06 1:30	33.06	92	1/1/06 1:30	239	6.2	0				
1/1/06 1:40	0	1/1/06 1:40	33.06	93	1/1/06 1:40	228	6.4	0				
1/1/06 1:50	0	1/1/06 1:50	33.06	93	1/1/06 1:50	241	5.5	0				
1/1/06 2:00	0	1/1/06 2:00	33.06	92	1/1/06 2:00	239	6.5	0				
1/1/06 2:10	0	1/1/06 2:10	33.14	91	1/1/06 2:10	245	5.1	0				
1/1/06 2:20	0	1/1/06 2:20	33.06	91	1/1/06 2:20	237	3	0				
1/1/06 2:30	0	1/1/06 2:30	32.89	93	1/1/06 2:30	226	3.9	0				
1/1/06 2:40	0	1/1/06 2:40	32.65	92	1/1/06 2:40	206	3.4	0				
1/1/06 2:50	0	1/1/06 2:50	32.65	93	1/1/06 2:50	218	4.8	0				
1/1/06 3:00	0	1/1/06 3:00	32.73	92	1/1/06 3:00	248	5.3	0				
1/1/06 3:10	0	1/1/06 3:10	32.65	93	1/1/06 3:10	242	4.3	0				
1/1/06 3:20	0	1/1/06 3:20	32.69	91	1/1/06 3:20	226	6.3	0				
1/1/06 3:30	0	1/1/06 3:30	32.73	90	1/1/06 3:30	234	2.6	0				
1/1/06 3:40	0	1/1/06 3:40	32.89	88	1/1/06 3:40	250	3.8	0				
1/1/06 3:50	0	1/1/06 3:50	32.73	88	1/1/06 3:50	258	7.1	0				
1/1/06 4:00	0	1/1/06 4:00	32.89	87	1/1/06 4:00	236	5.7	0				
1/1/06 4:10	0	1/1/06 4:10	32.81	89	1/1/06 4:10	232	3.7	0				
1/1/06 4:20	0	1/1/06 4:20	32.77	88	1/1/06 4:20	234	4.1	0				
1/1/06 4:30	0	1/1/06 4:30	32.77	89	1/1/06 4:30	223	6.9	0				
1/1/06 4:40	0	1/1/06 4:40	32.49	90	1/1/06 4:40	234	3.8	0				

Figure 2. An example single station input file for WEPPCLIFF. Input files follow a generic 'named' variable convention in which variable names are used to parse information. Names must be spelled identically to corresponding variables used in WEPPCLIFF (case insensitive).

Single File vs. Directory Processing

Input can also be provided when there is more than one station. This can be accomplished by providing each station separately as its own file (same convention as above) in a single directory. See the table of arguments for more details on how to operate WEPPCLIFF in directory processing mode.

OUTPUTS

Output Data Structure and Access

Internal Data Structure

Data in WEPPCLIFF is saved in a single R list called 'data'. Various functions in WEPPCLIFF source code read and save to this list during execution. If the export data flag is assigned a value 1-5 then some or all of this data structure will be saved in addition to the standard .cli file output. If no value is assigned or if a value of 0 is provided, only the standard output .cli file will be produced. There are up to 22 objects in this list depending on the options specified to WEPPCLIFF for each run.

Elements in the list are saved to specific locations (by index). You can access elements in this list with the following (LEFT: general data description; RIGHT: R subset command):

- **ORIGINAL INPUTS (AFTER FORMATTING / PREPROCESSING)**
  - input_precipitation_dataframe data[[1]]
  - input_alternative_dataframe data[[2]]
  - input_daily_dataframe data[[3]]
  - input_station_metadata_vector data[[4]]

- **WEPPCLIFF ALTERED INPUTS (QC / TRIMMING / CONVERSIONS / ETC)**
  - trimmed_qc_precipitation_dataframe data[[5]]
  - trimmed_qc_alternative_dataframe data[[6]]
  - trimmed_qc_daily_dataframe data[[7]]

- **STORM SEPARATION DERIVATIVES**
  - storm_breakpoints_list data[[8]]
  - storm_characteristics_dataframe data[[9]]

- **AGGREGATED TIMESERIES DATA AND STATISTICAL SUMMARIES**
  - aggregated_daily_timeseries_dataframe data[[10]]
  - aggregated_monthly_timeseries_dataframe data[[11]]
  - aggregated_yearly_timeseries_dataframe data[[12]]
  - aggregated_monthly_means_dataframe data[[13]]
  - annual_summary_output data[[14]]

- **GAP-FILLED VERSIONS OF PRIOR STRUCTURES**
  - gap_filled_precipitation_dataframe data[[15]]
  - gap_filled_storm_breakpoints_list data[[16]]
  - gap_filled_storm_characteristics_dataframe data[[17]]
  - gap_filled_daily_timeseries_dataframe data[[18]]
  - gap_filled_monthly_timeseries_dataframe data[[19]]
  - gap_filled_yearly_timeseries_dataframe data[[20]]
  - gap_filled_monthly_mean_dataframe data[[21]]
  - gap_filled_annual_summary_output data[[22]]

NOTE: Some of the above elements above may be empty (NULL) if various options are not specified, and all of them will be empty at some point during WEPPCLIFF execution.

Variable Headers and Units

WEPPCLIFF uses descriptive names for most variables stored in the primary data structure. There may be a few exceptions to this which are documented below. WEPPCLIFF also always uses metric units and any unit conversions that are applied are based on user specifications. A comprehensive list of all variable names and units are provided below:

NAME	UNIT	DESCRIPTION
AIR_TEMP	°C	air temperature
ANT	min	antecedent time
B_YR	year	beginning year (output)
BPS	integer	breakpoints
BTW	min	time between precipitation
CUM	min	cumulative time
D_CON	logical	depth connection logic
DEPTH_WM_INT	mm / hour	depth weighted mean intensity
DP_TEMP	°C	dewpoint temperature
DT_1	datetime	precipitation data datetime
DT_2	datetime	alternative data datetime
DT_3	datetime	daily data datetime
DUR	min	duration
ELEN	integer	number of events
ELEV	m	elevation
EMAX	integer	maximum SID
EMIN	integer	minimum SID
I30	mm / hr	maximum 30-minute intensity
INT	mm / hr	intensity
INT_AFT_30	mm / hr	30-minute intensity (lead looking forward)
INT_BEF_30	mm / hr	30-minute intensity (lag looking back)
LAT	DD	latitude
LON	DD	longitude
MAX_TEMP	°C	maximum temperature
MIN_TEMP	°C	minimum temperature
MM	month	month
OBS_YRS	years	observed years (input)
PDUR	min	storm precipitation duration
PEAKINT	mm / hr	peak storm intensity
PEAKTIME	decimal (0-1)	fractional time of PEAKINT
PRATIO	decimal (0-1)	ratio of PDUR to DUR
PRECIP	mm	precipitation
REL_HUM	%	relative humidity
SID	integer	storm identifier
SO_RAD	lang / day	solar radiation
STATION	name or ID or other	station identifier
SUM_AFT_TTH	mm	precipitation sum (lead looking forward)
SUM_BEF_TTH	mm	precipitation sum (lag looking back)

T_CON	logical	time connection logic
TIME_WM_INT	mm / hour	time weighted mean intensity
W_DIR	degrees from N	wind direction
W_VEL	m / s	wind velocity
YRS_SIM	years	total simulation years (output)
YYYY	year	year
YYYYMM	month	year month
YYYYMMDD	date	year month day
[METHOD]_EI	MJ-mm / ha-hr	erosion index (per method)
[METHOD]_KE	MJ / ha	kinetic energy (per method)
[NAME]_DEL	Δ variable	change since last observation (per variable)
[NAME]_QC	string	quality checking result (per variable)

Standard and Optional Outputs

WEPPCLIFF always produces a .cli file (short for climate file) which can be used as the climate input for a WEPP model simulation. This climate input file is documented in the WEPP user guide and is discussed in the 'Overview' section of this documentation.

Although the creation of a .cli file was the original purpose of WEPPCLIFF, now it is possible to export other data via data export functions. A summary of the data export flags, arguments, and products are included below:

FLAG	VALUE	PRODUCT	EXTENSION	TYPE
-ed	0	none	none	none
-ed	1	all internal data	.rds	binary
-ed	2	all internal data	.json	text
-ed	3	precipitation timeseries	.csv	text
-ed	4	daily timeseries	.csv	text
-ed	5	storm characteristics	.csv	text
-eb	t	precipitation breakpoints	.csv	text

Accessing Exported Output

The .rds file can be read by any machine running R with the following function: readRDS("filename.rds"). If the .json option is used in with the export data flag, then any language that supports .json reading will be able to open the output data structure, albeit there may be some additional formatting statements needed. The .rds is much more efficient in terms of storage, reading, and writing because it is saved in binary as opposed to human readable text. The other export options all save a specific dataframe from the primary data list as a .csv file (ideally for rapid analysis with a spreadsheet program such as Excel). These .csv files default to the gap-filled versions if that option is specified.

CONTRIBUTORS

Acknowledgements and Contact Information

Acknowledgements

As of the most recent release of WEPPCLIFF, only the lead developer has contributed to the development of the source code. Any contributions will be appropriately credited here as they are implemented. Those contributors who wish to have their contact information listed will be included below in order of contribution.

Within WEPPCLIFF there are several packages used from the R community. All of these important packages are listed here (in the order in which they are installed):

- backports
- crayon
- vctrs
- readr
- rlist
- iterators
- foreach
- doParallel
- EnvStats
- mice
- RcppParallel
- withr
- ggplot2
- profvis
- data.table

Citation for Published Works

McGehee, R.P., D.C. Flanagan, P. Srivastava. 2019. WEPPCLIFF: A command-line tool to process climate inputs for soil loss models. Submitted to Journal of Open Source Software on 22 November 2019. Available at: https://github.com/ryanpmcg/WEPPCLIFF

Contact Information

Ryan P. McGehee, Lead Developer
Email: rmcgehee@purdue.edu

ERRATA & UPDATES

Known Issues and Development Notes

Reported Issues

As issues become known, they will be documented here. Corresponding development notes are meant to address these issues and are documented below. Issues are bulleted and organized by version.

v1.3.1 (and earlier)

- Midnight precipitation in the first breakpoint position causes all subsequent breakpoints to be shifted down (incorrectly).

v1.2 (and earlier)

- When using very small amounts of data, some routines may be too strict on preventing execution from proceeding. It may be possible to increase WEPPCLIFF utility for small amounts of data.

v0.1

- The installation of R dependencies relied on some user interaction. It was requested that this process was more automated and still reliable across various systems.

Development Notes

This is a record of developments and developer notes to keep a coarse history of WEPPCLIFF source code development. Developments and notes are bulleted and organized by version.

v1.4

- Expanded and standardized code testing and tutorial.
- Added more precise control of time period trimming.
- Drastically improved memory utilization of large parallel runs (> 10x reduction).
- Drastically improved quality checking routines.
- Drastically improved graphical output functions.

v1.3.1

- Changed behavior of how missing metadata are filled.
- Fixed midnight precipitation shifting (first breakpoint).

v1.3

- Improved long term gap filling by tweaking gap detection algorithms.

v1.2

- Significantly improved gap detection and filling routines including the following:
    - Solar radiation is imputed first by multivariate imputation as opposed to univariate
    - Gap detection relies on outliers of empty month streaks and clustered dry periods
    - Precipitation time series imputation preserves intensity breaks better
    - Each stage of imputation has clustering recomputed for increased imputation ‘agility’

v1.1

- Modified library export to foreach calls for more stable gap filling in parallel instances.

v1.0.1

- Updated R package loading order to work with older Windows systems.
- Published a non-color version to comply with older Windows terminals.

v1.0

- Completed quality checking and gap filling routines.
- Published the public release of the code and documentation.

v0.5

- Added graphical output routines.
- Set foundation for quality checking and filling routines.
- Added input format checking. WEPPCLIFF now provides the names of missing variables if it does not detect them for either option [-alt] t/f.
- Changed the 'graphical output' option [-gd] to [-pd] due to conflicting argument parsing with Rscript.
- Fixed a minor bug impacting start and end dates.

v0.2

- Updated 30-minute intensity weighted averaging (it was not including times of no precipitation).
- Updated storm separation logic (some storms which were connected via e_con_aft = T were being wrongfully separated).
- Updated datetime conversion methods (some input files were being processed in seconds rather than minutes).
- Added Rscript.exe to the 'path' variable for Windows systems in the general installation procedure.
- Changed the function to install dependencies using 'binary' installation if possible.
- Changed the library for package installation and loading to a personal library which is created during the first time run with WEPPCLIFF.

v0.1

- Released WEPPCLIFF v0.1 to ARS NSERL lab for testing and refinement.