**AutoSTRM: Automated and Standardized River Monitoring**

Notes and Standard Operating Procedure

**BASICS**

The AutoSTRM system consists of 4 R scripts (clean.R, workup.R, update.R, graph.R). Together these files can be used to quickly obtain almost all data needed for reporting on stream restoration projects. There is no need to understand R or any other scripting languages—the user’s interaction with R is limited primarily to specifying filepaths and filenames.

**COMPONENTS**

*clean.R*

Accepts raw total station data. Returns two Excel files, one containing cross section data and the other profile data formatted for use in Meck spreadsheets (if desired), along with stationing. Each sheet also includes several form fields to be filled in by the user.

*workup.R*

Accepts any combination of cleaned XS data, cleaned profile data or pebble count data and creates new Excel spreadsheets that contain individual and aggregate statistics for each type of data input.

*update.R*

Accepts any combination of cleaned XS data, cleaned profile data or pebble count data and updates the corresponding database. If no database exists, the script will create a new database. New databases have several form fields to be filled in by the user.

*graph.R*

Accepts any combination of XS, profile or pebble count databases and creates the appropriate charts (as PDFs) for use in reporting.

**USE AND OVERVIEW**

*Setup:* Use of AutoSTRM requires an installation of R on your computer. The software is free, and can be found at [*https://www.r-project.org/*](https://www.r-project.org/). Once R is installed, the user may wish also install R Studio, an extended Graphical User Interface for R ([*https://www.rstudio.com/*](https://www.rstudio.com/)), though this is not strictly necessary. AutoSTRM makes use of the following R libraries: xlsx, gridExtra, RGraphics, gridBase, jpeg, Matrix. If the user does not have these libraries already, they can be installed by entering

* *install.packages("package\_name")*

into the R command line. This only needs to be done once.

*Schematic:* The point of AutoSTRM is to quickly extract useful information from raw stream survey data and pebble counts and produce useable graphics for reporting purposes. To do this, the scripts read in data from Excel, operate on the data, and then write the data to a new Excel file or PDF. The data can then be modified via Excel before the next script in the sequence is called. A general schematic of the AutoSTRM process is shown below.

**C:\Users\Randall.Jones\Desktop\Untitled Diagram.png**

*Figure 1. Generalized schematic for using AutoSTRM to extract charts and statistics from survey data.*

*Formatting:* Because AutoSTRM attempts to minimize user input in order to reduce error and increase throughput, the program expects the raw data to be formatted a certain way. Specifically, the initial input into clean.R must be a .xlsx file with four columns in the following order: Shot Number, Northing, Easting, Description. Furthermore the following character sequences in the description are protected: thw, ws, bkf, hc, ri, po, tob, ltob, rtob, xs, pro. That is, if they appear anywhere in the description then the program will use it to classify the shot. This can cause problems if, for example, you use “bridge” as a descriptor as the program will identify the shot as the beginning of a riffle (“bri”). To prevent this, either avoid using these character sequences out of their proper context or use the comment character “\_”; the program will ignore any text in the description after an underscore.

The program also expects shots to be consistently named, and to have a consistent delimiter (like = or -) to separate the shot sequence from the shot type, and the first portion of the name must have either “pro” or “xs” in it. Thus, a well-formatted description might look like “prot3-ws” or “xsref=tob”. You can also use multiple delimiters if desired, as long as the first delimiter separates the shot sequence from the shot type and is consistent for every shot. It is also important that profiles and cross-sections are given the same names year-to-year. If names are inconsistent then the data will be still be cleaned correctly but the program be unable to correctly update any existing databases.

Finally, the data should be collected (or reformatted) so that once you begin a shot sequence (e.g., xs2) all following points must be part of that sequence until the sequence is finished. The exception to this is if you cut up a continuous profile with cross sections (or vice versa, though that would be unusual.)

*Running R Scripts:*

*IF USING BASE R –* Go to *File>Open Script* and navigate to the fileyou want to run. In the section labeled *User Input*, change the program to match your needs (see “*Step-by-step”* for more information). In the R GUI go to *File>Source R Code* and again navigate to the file you want to run.

*IF USING R STUDIO –* Go to *File>Open File* and navigate to the file you want to run. In the section labeled *User Input*, change the program to match your needs (see “*Step-by-step”* for more information). Click on *Source* in the upper right corner of the upper left subwindow.

Note that when defining variables in R, text must be surrounded by single or double quotes.

*Step-by-step:*

**Data cleaning with clean.R**

1. Extract raw data from the data collector. Paste it into a .xlsx file, making sure the columns are separated correctly.
2. Open clean.R. If you haven’t installed the *xlsx* package, do so now. You may also want to install the other packages (gridExtra, RGraphics, gridBase, jpeg, Matrix) as they will be required later.
3. Scroll down until you see a section labeled *USER INPUT*.
4. Notice that there are five variables that you are asked to define
   1. mywd – Your working directory. This is the filepath to the folder that your raw data is in, and is also where the output will be written to
   2. datasource – The filename of your raw data (no extension needed, but the file must be .xlsx)
   3. outputname – The basis for the names of the program’s output. Two files will be written: [outputname]pro.xlsx and [outputname]xs.xlsx. The .xlsx will be appended automatically so DO NOT include that in outputname
   4. delimit – This is the delimiter you used to separate the shot sequence from the descriptors
   5. myr – The monitoring year. Can be anything you want (e.g., MYR-02, Baseline, etc.), but keep in mind that this will autopopulate form fields used to make charts later.
5. Define the variables, and run the script. The output will be in the same folder as the source file. Open output files to make sure the data was cleaned correctly.

**Extracting statistics with workup.R**

I.

1. Open up the profile output from datacleaner.R, which will be called [outputname]pro.xlsx.
2. Notice that the data is formatted so it can be used directly in the Meck spreadsheets, and that stationing is provided to make graphs by hand if desired.
3. Also notice that there are three blank columns in each tab:
   1. ri/ru/po/gl/st
      1. This column is used to specify the beginning and end of each stream morphology type (riffle, run, pool, glide, structure)
      2. Use either b or e and the morphology code (ri/ru/po/gl/st) to specify the beginning or end of a morphology. For example, bgl specifies the beginning of a glide while eri specifies the end of a riffle.
      3. When one morphology ends, another should begin. Thus, the end of a riffle that turns into a run should correspond to a cell that contains both eri and bru. This could look like eri/bru, eri-bru, eribru, brueri, etc. However, eri/bru or eri-bru is the preferred nomenclature.
      4. Structures (st) have the special scenario of *inclusion.* If a structure is included inside one continuous morphology, rather than existing between two distinct morphologies (including morphologies of the same type), then use the code IN to specify the inclusion. Thus, the beginning of an inclusion will look like bst-IN, and the end will be est-IN.
   2. Profile Name
      1. Simply fill out the cell immediately below this with the name of the profile. This is only used for naming other cells.
   3. Reach Membership
      1. Fill out the cell immediately below this with a numerical code that specifies the reach number. It is assumed that any given profile makes up no more and no less than one reach, so this code should be unique to each profile as well.

II.

1. Open up the XS output from datacleaner.R, which will be called [outputname]xs.xlsx.
2. There are 4 cells that need to be filled in each tab:
   1. Bankfull
      1. Enter the static bankfull height
   2. XS Name
      1. Enter a name for the XS
   3. Reach type
      1. Enter “p” for pool or “r” for riffle. If pool, FPA width and entrenchment ratio will not be calculated for the XS and the XS will not be included in aggregate data calculations
   4. Reach membership
      1. Enter a numerical reach code. This should correspond to the reach code for the profile that the XSs are in. There can be multiple XSs with the same reach code

You may also choose to edit the “description” column. The program automatically finds where the bankfull elevation intersects the collected survey data and uses this to calculates bankfull width and XS area. However, sometimes banks aggrade/degrade slightly, causing a premature or delayed intersection, significantly altering calculated statistics. If you wish to manually specify the location of the top of the left or right bank (or both), add the code “ltob” or “rtob” to the corresponding cells in the description column. It is recommended that tops of banks are recorded as “tob” during data collection in order to make this easy to do if the need arises. Note that specifying sufficiently wide banks will not alter the calculated XS area if the bankfull elevation intersects both sides anyway but will still alter the calculated width.

III.

1. Open up a blank pebble count template (pebble\_template.xlsx)
2. Make a new tab for each XS, keeping mind to name each tab the same name used to identify the pebble count/XS in previous years (if applicable)
3. Rename the file to whatever is desired
4. Fill in the missing data.
   1. Count - the pebble count for each size class
   2. XS Name - the name of the cross section that the count goes with
   3. Reach Membership - the reach code that the corresponds to the profile that the XS is in

IV.

1. Open workup.R. Define the directory that the source files are located in.
2. Notice the three logical variable workXS, workPro and workPart. When set to TRUE, the program will calculate statistics for XS, profile and pebble count data respectively. If set to FALSE, the corresponding data will be ignored (useful if you have already done the calculations or not collected the data yet).
3. Define the names of the appropriate source files, and the desired output names as well.
4. Run the script. The output will be in the same directory as the source files.

**Updating databases with update.R**

1. If you have not done steps I through III of “Extracting statistics with workup.R”, it is recommended to do this before running update.R. While not strictly necessary, this will give you an opportunity to review data and fill in form fields that will be used to automatically populate other form fields in the databases, particularly if a new database is being created.
2. Open update.R. Define the directory that the source files (and existing databases, if applicable) are located in.
3. Notice the three logical variable workXS, workPro and workPart. When set to TRUE, the program will update the XS, profile and pebble count databases respectively. If set to FALSE, the corresponding data will be ignored (useful if you have already updated the database or not collected the data yet).
4. Define the names of the appropriate source files, and the desired output names as well. If a file with the output name already exists, then the database will be updated. If it does not already exist, then the program will create a new database with this name in the same directory that the source files are in.
5. Run the script. If the program is updating an existing database, it will alert you when it does not find a match for any source data as well as if any parts of the database were not updated due to a lack of corresponding source data. Note that matches are made on the basis of equivalency between worksheet names in the source and target files.
6. Check the databases to make sure they were properly updated or created.

**Creating graphics with graph.R**

1. Evaluate each database (XS, profile and pebble) to make sure they have been properly updated, and the form fields have been filled out correctly
   1. XS Database
      1. Each worksheet corresponds to a different cross section. If the database is not new, then the second column (“XS Data”) will retain its data from the previous year, though several of the fields may need to be updated anyway. The data that goes in this column is as follows:
         1. Bankfull – static bankfull height for the cross-section. Autopopulated for new databases
         2. Graph Title/XS ID – Used as part of the graph title. Format is “[CROSS SECTION ID], [Riffle/Pool], [Reach]”
         3. Reach Type (p/r) – pool (p) or riffle (r). Affects data displayed and calculated on the graph. Autopopulated in new databases.
         4. Reach Membership – reach code that corresponds to the reach the XS is located on. Autopopulated in new databases
         5. Basin – river basin name
         6. Watershed – watershed name. Combined with Basin as part of the graph title
         7. Drainage Area – drainage area of the basin (in sq. miles)
         8. Image Code – a keyword used in the name of the cross section photos, used to locate the appropriate photo
         9. Image Folder – filepath to the location of the XS images
         10. Prep Date (as string) – the date the graph is being generated, formatted ‘MM/DD/YYY. Note apostrophe at the front of the date. This is used to format the date as a string.
         11. Crew – the people responsible for data collected
      2. Make sure that each monitoring year is appropriately defined, and note that the data from the rightmost monitoring year is used to calculate XS statistics, regardless of if it is the most recent data
   2. Profile Database
      1. Each worksheet corresponds to a different profile/reach. If the database is not new, then the second column (“Reach Data”) will retain its data from the previous year. If the database is new, there are two formfields to fill out:
         1. Site name – name of the project site
         2. Reach – What reach is this? Does not have to be a numeric reach code, could be e.g., T1
      2. Make sure each monitoring year field is appropriately filled out. Also notice the three rightmost columns: “Cross-Sections”, “Structures” and “Adjustment”. Filled out the cross-sections and structures columns with the elevations of the cross-sections and structures next to their corresponding station. The adjustment column takes a single real number immediately below the header that is used to shift the profile downstream (positive number) or upstream (negative number) in order to align profiles from different years. By default it is set to 0.
   3. Pebble database
      1. Each worksheet corresponds to a different cross section. If the database is not new, then the second column (“XS Data”) will retain its data from the previous The data that goes in this column is as follows:
         1. Site Name – name of the project site
         2. XS Type (p/r) – used in creating the graph title
         3. Membership – numeric reach membership. Autopopulated in new databases
         4. XS Name – Used in the graph title.
2. Once all databases are ascertained to be correctly updated, open graph.R. Define the directory that the source files are located in.
3. Notice the three logical variable workXS, workPro and workPart. When set to TRUE, the program will create charts for the XS, profile and pebble count data respectively. If set to FALSE, the corresponding data will be ignored (useful if you have already produced the figures or not collected the data yet).
4. Define the names of the appropriate source databases. The names of the output files will be automatically generated based on information in the databases.
5. Run the script. The figures will be produced as PDFs in the same directory that the source data is located.

*Common Issues*:

* *General*
  + File paths can be specified using either forward (/) or back (\) slashes, but if backslashes are used you must escape it with another backslash (\\) as the backslash itself is a special character in R.
* *clean.R*
  + Using protected descriptors inappropriately (e.g., having a point with “bridge” as a designator will mark it as begin riffle unless commented out with “\_”)
  + Failing to change a shot name (e.g., having two eri shots with no bri between)
  + If the error

Error in .jcall(wb, "Lorg/apache/poi/ss/usermodel/Sheet;", "createSheet", : java.lang.IllegalArgumentException: The workbook already contains a sheet of this name

is given, this means that you are trying to write over a file you have already generated. Either delete the old output, or change the output name in the script. Then, rerun the script.

* *workup.R*
  + *Coming soon*
* *update.R*
  + *Coming soon*
* *graph.R*
  + If you delete a column of data from a database, then you must delete the column itself (not the only the data). Excel “remembers” if data populated a column, so when R reads in the spreadsheet it will read in a column of NA values. This causes problems because graph.R bases some calculations (like the number of years of data) on the number of columns read in.

*Figures:*

*How the scripts work (for advanced troubleshooting and modification)*: