

Southeast Coast Network

National Park Service
Inventory and Monitoring Division



Standard Operating Procedure 2.2.14

Wadeable Stream Data Processing and Analysis—Version 1.0

Please cite this as:

McDonald, J. M., E. N. Starkey, and M. B. Gregory. 2018. Wadeable stream data processing and analysis—Version 1.0. Southeast Coast Network Standard Operating Procedure NPS/SECN/SOP—2.2.14. National Park Service, Athens, Georgia.

Summary

The following standard operating procedure (SOP) describes how to process and analyze the data collected at the reaches chosen as part of the *Protocol for Monitoring Wadeable Stream Habitat Conditions in Southeast Coast Network Parks*. This SOP outlines the basic steps that should be followed to create tables and figures describing each surveyed reach as well as provides instructions for processing the stream survey data collected using the Trimble S6 robotic total station and the TSC3 data collector. The steps required to convert the basin characteristics data into report-ready tables are also described in this SOP. Initially, all of these data will be processed using Microsoft Excel and/or ArcMap 10.x; though these procedures will be updated as data entry and reporting needs are finalized and a formal database is developed based on those needs. Lastly, this SOP provides a description of the statistical tests that are used to define and assess the statistical significance of status, change, and trends in the survey data. It is assumed that the user is familiar with Microsoft Excel, ArcMap 10.x, and the statistical software R.

Revision Log

Revision Date	Author	Changes Made	Reason for Change	New Version #

Contents

	Page
Summary	1
Revision Log	1
Required Materials	3
Overview	3
Procedure	4
Data Processing	4
Reach Characteristics	4
Standard Transects.....	5
Basin Characteristics	6
Detailed Transect Cross-Sections.....	7
Longitudinal Profile.....	10
Stream Overview Map.....	11
Data Analysis.....	13
Status	13
Change.....	13
Trends	14
Literature Cited	21
Appendix A. R Code to Create LULC Table.....	22
Appendix B. R Code to Extract Cross-Sectional Data.....	24
Appendix C. R Code to Plot Longitudinal Profile	39

Required Materials

- Computer
- Certified stream survey data
- Shapefiles of watershed area attributed with basin characteristics
- Shapefiles with detailed transect survey data
- Microsoft Excel*
- ArcGIS 10.X*
- R statistical software (base install)*

*Any use of trade, firm, or product names is for descriptive purposes only and does not imply endorsement by the U.S. Government.

Overview

This standard operating procedure (SOP) describes the steps that should be followed to process the data collected as part of the *Monitoring Wadeable Stream Habitat Conditions in Southeast Coast Network Parks: Protocol Narrative* (McDonald et al. 2018a). This SOP also outlines the procedure for exporting these data as tables and figures that summarize the condition of the in-stream and riparian habitat of the surveyed streams. These tables and figures will be used in the data reports produced by the Southeast Coast Network as described in SOP 2.2.15 *Wadeable Stream Reporting Guide—Version 1.0* (McDonald 2018b). Additionally, this SOP describes the data analysis techniques that will be used to determine if there are any spatially- or temporally-explicit changes or trends in stream habitat conditions in SECN wadeable streams.

The first portion of this SOP describes the steps that should be followed to process the reach and standard transect data that were certified following the procedures described in SOP 2.2.13 *Wadeable Stream Data Entry and Storage—Version 1.0* (McDonald 2018c).

These certified data can be found on the SECNshare drive at the following location:

SECNshare: \Monitoring\StreamHabitat\Wadeable\Data*PARK*\C_Certified*PARK*###

The second portion of this SOP outlines the steps that should be followed to convert each reach's geospatial data (basin characteristics, detailed cross-sections, and longitudinal profile) into tabular form. Once in tabular form, these data will be used to create figures/tables that describe upstream basin characteristics, characterize the morphology of the detailed cross-sections, and describe the longitudinal profile (slope) of the surveyed reach.

The geospatial data can be found on the SECN GIS drive at the following location:

The third and final part of this SOP describes how to analyze the stream survey data. While these techniques will likely be updated once multiple surveys have been completed, the initial surveys will allow the relationship between stream habitat conditions and upstream basin characteristics to be determined. Besides providing a baseline from which future change can be measured, this analysis will allow for the identification of heavily impacted streams and provide resource managers with useful information regarding the health of their wadeable streams.

Initially, all of these procedures will be done manually using Microsoft Excel, ArcMap 10.x, and R statistical software. The standard data summarization procedures related to the tabular data will be automated once data entry and reporting needs are sufficiently finalized to inform back-end database development.

Procedure

Data Processing

Reach Characteristics

These data (from Form 1) are found on the SECNshare drive in each reach's certified data folder and are named based on when the survey was done. As an example, the reach characteristics of CHAT reach 001 surveyed on 20 September 2016 and certified by staff with initials 'JMM' would be named:

- CHAT001_20160920_F1_cert_JMM.xlsx

The data in this workbook include survey conditions, geomorphic channel units (GCU) percentages, the results of the pebble count, large woody debris (LWD) counts, and canopy cover data from the densiometer readings. The Form 1 'Data Entry and Processing Template' workbook used in data entry (see SOP 2.2.13) will also auto-populate tables that will be used in the annual reports. The data processing-related tabs in the workbook take the data referenced in each data entry tab and calculate descriptive statistics. Also contained within the Form 1 Data Entry and Processing Template workbook are informative tabs identifying the specific equations used to populate each table in case the equations need to be adjusted or re-entered in the future, and for reference during back-end database development.

The annual report tables that are created from the reach characteristics data are:

- Table 4—Geomorphic Channel Units (Percent coverage of riffles, runs, glides, and pools)
- Table 5—Pebble Count statistics (Summary statistics of pebble count survey)
- Table 6—Large Woody Debris (Counts and description of LWD found within reach)
- Table 7—Large Woody Debris summary statistics

- Table 8—Canopy Cover (Transect- and Reach-scale estimates of canopy cover)

The only table that needs to be manually updated is Table 6—LWD. Depending on the number of pieces of LWD that were recorded, the number of rows will change in Table 6. Use the example code from the Table 6—Equations tab to fill in the table. If any of the tables were not auto-populated, follow the formula to determine the problem or re-enter the equation using the example from each table's equation tab.

Once the tables have been checked by the protocol lead, they can be copied and pasted into the data reports. For instructions on how to construct these reports see SOP 2.2.15 (McDonald et al. 2018b).

Standard Transects

Standard transect data (from Form 2) are found on the SECNShare drive in each reach's certified folder and are named based on the date of the survey. As an example, certified reach characteristics of CHAT reach 001 surveyed on 20 September 2016 by staff with initials 'JMM' would be named:

CHAT001_20160920_F2_cert_JMM.xlsx

The data included in this workbook include all of the width, depth, and bank characteristic data collected at each standard transect.

Similar to the Reach Characteristics tables, the workbook used to enter the standard transect data (Form_2_DataEntryProcessingAndReporting_Template.xlsx) will auto-populate tables that can be copied and pasted into the data reports. The equations that were used to populate the tables are also included in separate tabs within the data entry/processing/and reporting template.

The annual report tables that are created from the standard transect data include:

- Table 9—Thalweg Characteristics (Depth and Position)
- Table 10—Width Averages and Variability (Wetted, active channel, bankfull, and channel-full widths)
- Table 11—Bank Characteristics (Bank height, floodplain height, channel-full height, water depth at base of bank, undercut, bank angle, vegetation cover, and bank stability)

All characteristics are reported for both left and right banks individually.

Tables 9 and 10 provide summary statistics (min, max, mean, standard deviation, and coefficient of variation) of the different thalweg and width measures. Bank characteristics need to be processed before they can be reported. Bank characteristic processing is automated within the entry/processing/and reporting template but they will be briefly explained here in case the data need to be manually calculated.

How to calculate bank heights

1. First, bank heights are standardized by subtracting water depth at base of bank from bank height (new field called 'height above water surface'). Bank heights need to be standardized

because the bases of some of the banks are emergent and some are beneath the water's surface.

2. After height above water surface has been calculated, the height above thalweg is determined by adding thalweg depth to the bank height above water surface.
3. Once height above thalweg has been calculated, the Processing.Bank_1_HeightAbove tab queries the Entry.Data floodplain width columns to determine floodplain presence/absence and will then fill in the Floodplain Height (Bankfull) and Channel-full heights above thalweg columns. All of these data are then imported into the Processing.Bank_3_Reporting tab.

How to calculate bank stability

1. Bank stability index scores are calculated using bank angle, vegetative cover, and sediment type from the Entry.Data tab and bank height above thalweg from the Processing.Bank_1_HeightAbove tab.
2. The scores are added together, a transect-scale score is calculated, and each bank is classified, based on the score, as being 'stable', 'at-risk', 'unstable', or 'very unstable' (from Fitzpatrick et al. 1998).

Make sure all of the standard transect tables are correctly auto-populating and that the data make sense. If the tables are not auto-populating, follow the formula to determine the problem or re-enter the equation using the example from each Table's equation tab. Tables can be copied and pasted into the data reports after they have been checked/certified by the protocol lead. For instructions on how to construct the reports see SOP 2.2.15 (McDonald et al. 2018b).

Basin Characteristics

The basin characteristics of the watershed draining to each surveyed reach were calculated following the procedures described in SOP 1.2.15 *Wadeable Stream Basin- and Segment-Scale Data Summary—Version 1.0* (McDonald et al. 2018d). All of the geomorphic characteristics of each watershed are contained in the *PARK###_WS.shp* within the *PARK###* specific folder within the Site folder.

These data can be found on the SECN GIS drive at the following location:

SECNgis: \GIS\projects\StreamHabitat\Wadeable\PARK\Data\Site\PARK###

Use the following steps to create the basin characteristics table:

1. Open ArcCatalog and navigate to the appropriate park's geodatabase. Preview the *PARK###_WS* table and export the table to the A_RawData folder (SECN Share:\NEW DIRECTORY STRUCTURE\Monitoring\StreamHabitat\Wadeable\Data\PARK) as a .txt file named *PARK###_BasinCharacteristics.txt*.
2. Open the exported table in Microsoft Excel and copy and paste the data into the *BasinCharacteristics_DataEntryProcessingAndReporting_Template* (which can be found on

the SECNshare drive: \Monitoring\StreamHabitat\Wadeable\Data_templates). Enter the park name and site number (do not pad site number with zeros) on this sheet.

3. This workbook will populate reach-scale report Table 2—Basin Characteristics. Pay attention to the column headings when you paste the data as the exported basin characteristics table may have the variables in a different order depending on the order the basin characteristics were calculated.
4. Check the table to make sure that the values make sense and then save the workbook to the C_Certified folder for the park and name the file:

PARK###_YYYYMMDD_BasinCharacteristics_CERT_JMM.xlsx

In order to produce the land use/land cover (LULC) table, use the R code from Appendix A or found on the SECNshare drive at:

SECNShare: \Monitoring\StreamHabitat\Wadeable\Code (named “Combine_LULC.r”)

This code will combine the LULC data and export one table as a comma separated file type (.csv file). This report table (Table 3 of the reach-scale reports) can be copied and pasted into the Basin Characteristics processing template. Check to make sure that the LULC table was properly populated and then reference SOP 2.2.15 (McDonald et al. 2018b) for instructions on how to input the table into the reach-scale reports.

Detailed Transect Cross-Sections

The following section describes how to process the data that were collected on the detailed transects using the total station. The data will first be verified and formatted in Microsoft Excel and then run through R to calculate the characteristics of each cross-section. The last step is to take the data calculated in R and format them for use in the data reports.

The detailed transect cross-sectional survey data (collected using the total station) can be found on the SECN GIS drive at the following location:

SECN GIS: \GIS\projects\StreamHabitat\Wadeable\PARK\Data\Site\PARK###

Each survey will be named based on the day of the survey. As an example, cross-sectional data surveyed at CHAT site 001 on 20 September 2016 by staff with initials ‘JMM’ would be named:

CHAT001_20160920_Detailed_JMM

If the survey data have not been uploaded to the GIS drive see SOP 2.2.13 for detailed instructions on how to export the data from the TSC3 handheld.

Once the data have been located, in ArcCatalog export the attribute table of the detailed survey as a .txt file, and save the file on the SECNshare drive in the following location:

SECNshare drive: \Monitoring\StreamHabitat\Wadeable\Data\PARK\A_RawData\PARK###

Name the file using the same example as above (and exported by staff with initials 'JMM'):

CHAT001_20160920_Detailed_raw_JMM.txt

Use the following steps to create figures to visualize the surveys and check to make sure all of the points are labeled correctly and all of the diagnostic points were correctly identified.

1. Open Microsoft Excel
2. Open the .txt exported from ArcCatalog
 - a. You will need to change the dialog box in the lower right corner of the Open file window to read "All files" to see the .txt.
3. Copy the data for each transect (and longitudinal profile) into their own tab (name the tab according to the transect) and rename the original tab "*PARK###_raw_ YYYYMMDD*"
4. Arrange the points so that the left bank benchmark is the first row of data (Figure 1).
5. In each transect tab, create a column named "d2b" (distance to benchmark)
6. Use the Pythagorean Theorem to calculate the distance between each point and the benchmark by converting the differences in Northings and Eastings from one point to another into distances (Figure 2).
 - a. $=\text{SQRT}(\text{POWER}(D\$2-D3,2)+\text{POWER}(E\$2-E3,2))$
7. Create another column called XS_dist and populate it by multiplying all of the LBMOC point d2b values by -1 and then simply copy all of the d2b values for all of the other points (Figure 3).
 - a. This is done because the left benchmark is used to standardize the plots and all future change will be measured from this point.
8. Plot the cross-sections on a scatter-plot (first, copy and paste (values) the XS_dist column and then cut and paste the benchmark row between the last LBMOC point and the first LOC point (Figure 4) so that the data plots correctly).
9. Plot each of the diagnostic points that were surveyed to make sure that they are labelled correctly and are in the correct position (i.e., channel-full points, water's edge points, and bankfull points). When checking and verifying the data use Form 3 and the standard transect data to help interpret the cross-section.
10. Add two new fields, (1) named "Code", and (2) named "Position." For both fields, copy and paste all of the original point codes into these fields and re-label the original "Code" field as "Code_raw." When checking the original point locations, if one or more points need to be relabeled, do so in the new "Code" field. Figure 5 is an example of two cross-sections and the

process that should be followed to verify the data. If any of the left of benchmark points is a diagnostic point (e.g., channel full on the river left is left of the benchmark), relabel the point LBMOC in the “Position” field (this is done for processing reasons).

- a. In the top cross-section on Figure 5, all of the points are accurately positioned and because of the entrenched nature of the stream along this transect, there will be no bankfull points.
- b. In the bottom cross-section, the channel-full bank tops and the points corresponding to the bankfull top of bank and floodplain margin had to be relabeled.
 - i. The channel-full bank tops had to be relabeled because they need to be on a flat area of the bank (may be difficult to accurately pin point in the field due to vegetation).
 - ii. The bankfull bank top point also needs to be on the closest flat point on the floodplain.

11. Additionally, make sure that the benchmarks are labeled (in the “Code” field) as LBM and RBM (not just BM). This will be important for when these data are processed in R.

Once the cross-sections have been checked and the points have been properly label and verified (by the Protocol lead), copy and paste the Name, Code (new field), Northing and Easting Coordinates, and Elevation of each transect into their own Microsoft Excel workbook (Figure 6) and save the file on the SECNshare drive at the following location:

SECNshare drive: \Monitoring\StreamHabitat\Wadeable\Data\PAK\C_Certified\PAK###

Name the file using the same example as above:

CHAT001_20160920_T#.csv

The next step is to calculate the cross-sectional characteristics in R. Use the code from Appendix B or found on the SECN share drive at:

SECNShare:\ Monitoring\StreamHabitat\Wadeable\Code (named
“Extract_CrossSectional_Data.r”)

This code will calculate the bankfull and channel-full characteristics of the three cross-sections and export a .csv file for each cross-section containing these data. Figures will also be exported illustrating the location of the diagnostic points and the areas that were used to calculate the bankfull and channel-full characteristics of each cross-section. These will be Figures 7 through 9 in the annual data reports (see SOP 2.2.15). Additionally, a figure comparing the three detailed transects will also be created that will be used as Figure 10 in the annual data report.

The last step that needs to be taken to make the data tables report-ready is to copy and paste the data from the individual .csv files into the CrossSection_DataTable_template.xlsx found on the SECN share drive at:

SECN Share:\NEW DIRECTORY

STRUCTURE\Monitoring\StreamHabitat\Wadeable\Data_templates

Check to make sure that the tables (Tables 12–15) were properly populated and then save the workbook to the C_Certified folder for the reach and name the file:

PARK###_YYYYMMDD_XS_tables.xlsx

Longitudinal Profile

The data needed to calculate the longitudinal profile can be found in the detailed transect attribute table that was exported in the previous step and can be found on the SECNShare drive in the following location:

SECNShare drive: \Monitoring\StreamHabitat\Wadeable\Data\PARK\A_RawData\PARK###

As an example, data surveyed at CHAT site 001 on 20 September 2016 by staff with initials ‘JMM’ would be named:

CHAT001_20160920_Detailed_raw_JMM.xlsx

Use the following steps to process the data and create the longitudinal profile:

1. If it hasn't been done already, copy the points labeled LONG from the “raw data_YYYYMMDD” tab and paste it into a new tab named “Longitudinal Profile”
2. Arrange the data in “Longitudinal_profile” so that most upstream point (WS11) is first and the most downstream point (WS01) is last (Figure 7).
 - a. Also place any riffle- and step-top points between the appropriate transects (e.g., a riffle-top was surveyed between T5 and T6 and was named WS56. Place this point between WS5 and WS6).
3. Use the Pythagorean Theorem to calculate the distance between points by converting the differences in Northings and Eastings from one point to another into distances.
 - a. $=\text{SQRT}(\text{POWER}(D2-D3,2)+\text{POWER}(E2-E3,2))$
4. Plot the points with distance between points on the x-axis and elevation on the y-axis.
5. Remove obvious outliers that do not conform to the downstream trend of decreasing elevation

6. Once the data have been quality controlled, copy and paste the “Name”, “Northing”, “Easting”, and “Elevation” columns into a new worksheet, name it “PARK###_YYYYMMDD_Long.csv”, and save it to the reach-specific C_Certified folder.

Once the longitudinal profile has been formatted and saved to the C_Certified folder, use the R Code in Appendix C or in the code folder named “Plot_Longitudinal_Profile.r” to plot the longitudinal profile and create a report ready figure (Figure 11 in the annual data reports) depicting the longitudinal profile.

Stream Overview Map

The Reach Map (Form 4) and survey data will be used to create this illustrated representation of the surveyed reach. The purpose of these maps is to provide a visual context for readers that will help them understand the data that are presented in the reports. While these maps are not ‘to scale’ there should be an attempt to keep an equal distance between transects and to draw the reach so that, relative to each other, transect widths are depicted reasonably accurately. These maps should give the user an understanding of gross variability and occurrence/absence of important geomorphic surfaces and in-channel features and habitats (Figure 8).

Use the following steps to create the stream overview map:

1. Take the scanned map (which can be found in the B_Scanned-Forms folder on the SECNshare drive) and import it into either Adobe Illustrator or Microsoft Powerpoint.
 - a. While digitizing and annotating the map is much easier in Illustrator, the steps can also be done in Powerpoint.
2. Draw a hollow polygon that is 6.5×6.5 inches (16.5×16.5 centimeters [cm]). Scale the map so that it fits within the polygon.
3. Digitize the stream channel and add in the benchmarks and transects.
4. Next, use the tables from Form 2 to add in the channel-full bank tops, undercuts and floodplains.
 - a. The channel-full bank top will be at the same position as the stream channel when there is no floodplain and will be to the left or right when there is a floodplain.
 - b. Channel-full bank top will overlap into the channel when undercutting is present.
5. Once the channel-full bank tops are drawn in label the undercut areas and floodplain.
6. Using the tables from Form 1, add in
 - a. The dominant geomorphic channel units between transects
 - i. If 75% riffle put an R between transects,

- ii. If 60% riffle and 40% pool put both R and P
 - b. The occurrence of in-channel bars (as grey triangles), and
 - c. The location of large woody debris.
7. Next, add in canopy cover indicators at each transect (coloring them black and then adjusting transparency depending on % cover).
 8. Finally, add in a legend, label any important out-of-channel features (i.e., hillslopes or historical terraces), and indicate the direction of flow.

When you have completed the digitizing, have the survey crew provide edits/comments and modify the figure as needed. Once the map has been edited, finish the map by providing a label indicating which reach (site) is being shown and the date of the survey. Save the finished map to:

SECNShare:\ Monitoring\StreamHabitat\Wadeable\Reports\ *PARK* \Figures

and name the file:

PARK###_YYYYMMDD_SiteMap.ai (or .xlsx)

The date should be the date of the survey. The stream overview map will be Figure 6 in the annual data reports (see SOP 2.2.15).

Data Analysis

The data collected at each reach are used to determine: (1) the status of each stream reach; (2) whether detectable change occurred at each stream reach between sample intervals; and (3) whether there are detectable trends in the surveyed variable at each stream reach through time. Prior to any analysis, the normality of the data will be assessed. While transformation of data to a normal distribution will allow for parametric tests to be used, interpretation of the results of these tests is potentially difficult. Transformations should be used with caution. A working document detailing how each variable will be used to characterize the status of each reach and detect change and trends is available on:

SECN share drive:\NEW DIRECTORY
STRUCTURE\Monitoring\StreamHabitat\Wadeable\Data

and is named:

WADE_Variable_Purpose_and_Analysis_YYYYMMDD.xlsx

Status

Summary statistics of reach-averaged measurements are used to characterize the reach-scale status of the surveyed reaches. Standardized reach-scale metrics will be used to facilitate comparisons between all of the surveyed streams within the network. These metrics will be plotted to visually compare reaches across the network, by physiographic province, and by Strahler stream order (Figure 9). Statistical significance of differences between paired sites (e.g., KEMO 001 and KEMO 002) at the transect-level is tested using a two sample *t*-test or Fisher exact test (for proportion data), or the Wilcoxon rank sum test (if the data are non-normal). These methods will be repeated every sampling interval and will be included in each network-level synthesis report. Information on which reaches were chosen as paired sites is available in the reach suitability report for each park. Reports are available on IRMA or by request from the Southeast Coast Network.

Change

After the second and all subsequent sampling interval(s), change analyses will be conducted. The objective of these analyses is to determine if there has been a significant change (in means or variance) in the surveyed variables between two sampling intervals (i.e., from time 1 to time 2 as well as from time 1 to time *n*). Changes in channel widths, bank heights, and bed sediment distributions (pebble count) between sampling intervals at each reach are evaluated using a paired *t*-test if the variable is normally distributed, the nonparametric Wilcoxon signed rank test if it is non-normal, or a Fisher exact test (for proportion data). Similar to the Student's paired *t*-test, the Wilcoxon signed rank test determines whether the means of two paired samples are significantly different while not requiring the data to be normally distributed (Crawley 2005). Additionally, the *F*-test (assumes normality) or Levene's test (does not require normality) will be used to determine whether the variance in thalweg position and depth are equal between sampling intervals (difference in variance is more informative for habitat quality than difference in means). Lateral migration rates and differences in the cross-sectional areas of the detailed transects will be quantified by comparing the total station surveys.

Trends

After each reach has been surveyed at least four times (minimum number needed for trend analysis), linear mixed models, linear or multiple regression, or the Mann-Kendall test will be used to determine whether there is a detectable trend in reach conditions. Due to the repeated measures (transect) design, linear mixed models will be used to determine whether trends are present at the reach-scale as well as at the transect-scale. Regression models will be used to determine if reach-averaged conditions are changing through time. If the data are not normally distributed, a nonparametric technique such as the Mann-Kendall test will be used to evaluate whether there is a detectable trend. The Mann-Kendall test uses Kendall's Tau to determine if there is a statistically significant monotonic (upward or downward) trend in the data (Helsel and Hirsch 2002).

Additionally, groups of three or more reaches (e.g., all reaches within a park or all reaches within a management unit or habitat type) may be compared using linear mixed models or the regional Kendall test to determine if similar trends are apparent within these groups of sites. Potential causes of any trends will be explored by analyzing land cover data to determine if any major changes (e.g., large-scale conversion of forest to development) may be contributing to the observed trend. These potential drivers of change can be incorporated into the mixed effect and in the regression models as covariates. Available climatological and hydrological data also will be analyzed to determine if large flood(s) influenced the reach.

HOBE_01_2016-07-26_Detailed_raw_JMM.xlsx						
	A	B	C	D	E	F
1	OBJECTID	Name	Code	Northing	Easting	Elevation
2	62	BM1	BM	3647833	617397.7	178.4433
3	73	1001	LBMOC	3647837	617387	179.8654
4	74	1002	LBMOC	3647837	617387.4	179.5951
5	75	1003	LBMOC	3647837	617388	178.969
6	76	1004	LBMOC	3647837	617388.3	178.8542
7	77	1005	LBMOC	3647837	617389.1	178.3616
8	78	1006	LBMOC	3647836	617390.1	178.1434
9	79	1007	LBMOC	3647836	617391.4	178.2956
10	80	1008	LBMOC	3647836	617392.2	178.4152
11	81	1009	LBMOC	3647835	617393.7	178.4228
12	82	1010	LBMOC	3647835	617393.7	178.4208
13	83	1011	LBMOC	3647835	617394.6	178.4107
14	84	1012	LBMOC	3647834	617395.5	178.3762
15	85	1013	LBMOC	3647834	617396.4	178.3526
16	86	1014	LOC	3647833	617399.6	178.3476
17	87	1015	LOC	3647832	617400.9	178.1973
18	88	1016	LOC	3647832	617402.5	178.2122
19	89	1017	LBT	3647832	617403.5	178.2603
20	90	1018	LOC	3647832	617403.7	178.2101

Figure 1. Each transect (and the longitudinal profile) was copied and pasted into its own tab. The benchmark corresponding to the transect was input into the first row of data.

HOBE_01_2016-07-26_Detailed_raw_JMM.xlsx						
	A	B	C	D	E	F
1	OBJECTID	Name	Code	Northing	Easting	Elevation d2b
2	62	BM1	BM	3647833	617397.7	178.4433
3	73	1001	LBMOC	3647837	617387	179.8654
4	74	1002	LBMOC	3647837	617387.4	179.5951
5	75	1003	LBMOC	3647837	617388	178.969
6	76	1004	LBMOC	3647837	617388.3	178.8542
7	77	1005	LBMOC	3647837	617389.1	178.3616
8	78	1006	LBMOC	3647836	617390.1	178.1434
9	79	1007	LBMOC	3647836	617391.4	178.2956
10	80	1008	LBMOC	3647836	617392.2	178.4152
11	81	1009	LBMOC	3647835	617393.7	178.4228
12	82	1010	LBMOC	3647835	617393.7	178.4208
13	83	1011	LBMOC	3647835	617394.6	178.4107
14	84	1012	LBMOC	3647834	617395.5	178.3762
15	85	1013	LBMOC	3647834	617396.4	178.3526
16	86	1014	LOC	3647833	617399.6	178.3476
17	87	1015	LOC	3647832	617400.9	178.1973
18	88	1016	LOC	3647832	617402.5	178.2122
19	89	1017	LBT	3647832	617403.5	178.2603
20	90	1018	LOC	3647832	617403.7	178.2101

Figure 2. Calculating the distance between each point and the benchmark on river left. The \$ sign is used so that the formula can be auto-filled down the column (so that the distance being calculated is from each subsequent cell to the first cell).

HOBE_01_2016-07-26_Detailed_raw_JMM.xlsx								
	A	B	C	D	E	F	G	H
1	OBJECTID	Name	Code	Northing	Easting	Elevation	d2b	XS_dist
2	62	BM1	BM	3647833	617397.7	178.4433	0	0
3	73	1001	LBMOC	3647837	617387	179.8654	11.43815	-11.4381
4	74	1002	LBMOC	3647837	617387.4	179.5951	10.9823	-10.9823
5	75	1003	LBMOC	3647837	617388	178.969	10.41417	-10.4142
6	76	1004	LBMOC	3647837	617388.3	178.8542	10.04929	-10.0493
7	77	1005	LBMOC	3647837	617389.1	178.3616	9.218737	-9.21874
8	78	1006	LBMOC	3647836	617390.1	178.1434	8.08806	-8.08806
9	79	1007	LBMOC	3647836	617391.4	178.2956	6.842165	-6.84216
10	80	1008	LBMOC	3647836	617392.2	178.4152	5.984005	-5.98401
11	81	1009	LBMOC	3647835	617393.7	178.4228	4.36563	-4.36563
12	82	1010	LBMOC	3647835	617393.7	178.4208	4.368609	-4.36861
13	83	1011	LBMOC	3647835	617394.6	178.4107	3.300902	-3.3009
14	84	1012	LBMOC	3647834	617395.5	178.3762	2.368095	-2.3681
15	85	1013	LBMOC	3647834	617396.4	178.3526	1.323479	-1.32348
16	86	1014	LOC	3647833	617399.6	178.3476	1.974753	1.974753
17	87	1015	LOC	3647832	617400.9	178.1973	3.327497	3.327497
18	88	1016	LOC	3647832	617402.5	178.2122	4.970131	4.970131
19	89	1017	LBT	3647832	617403.5	178.2603	6.031689	6.031689
20	90	1018	LOC	3647832	617403.7	178.2101	6.230958	6.230958

<

Figure 3. Distance along the transect is calculated by either copying the number from the d2b field or multiplying by -1 for all of the LBMOC points.

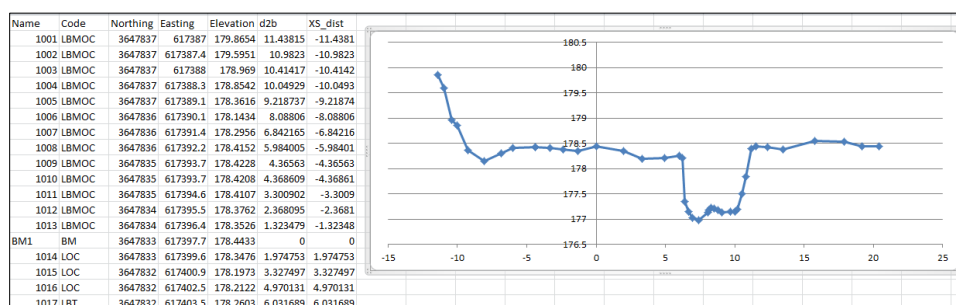


Figure 4. The benchmark point was moved between the last LBMOC point and the first LOC point so that the cross-section would plot correctly.

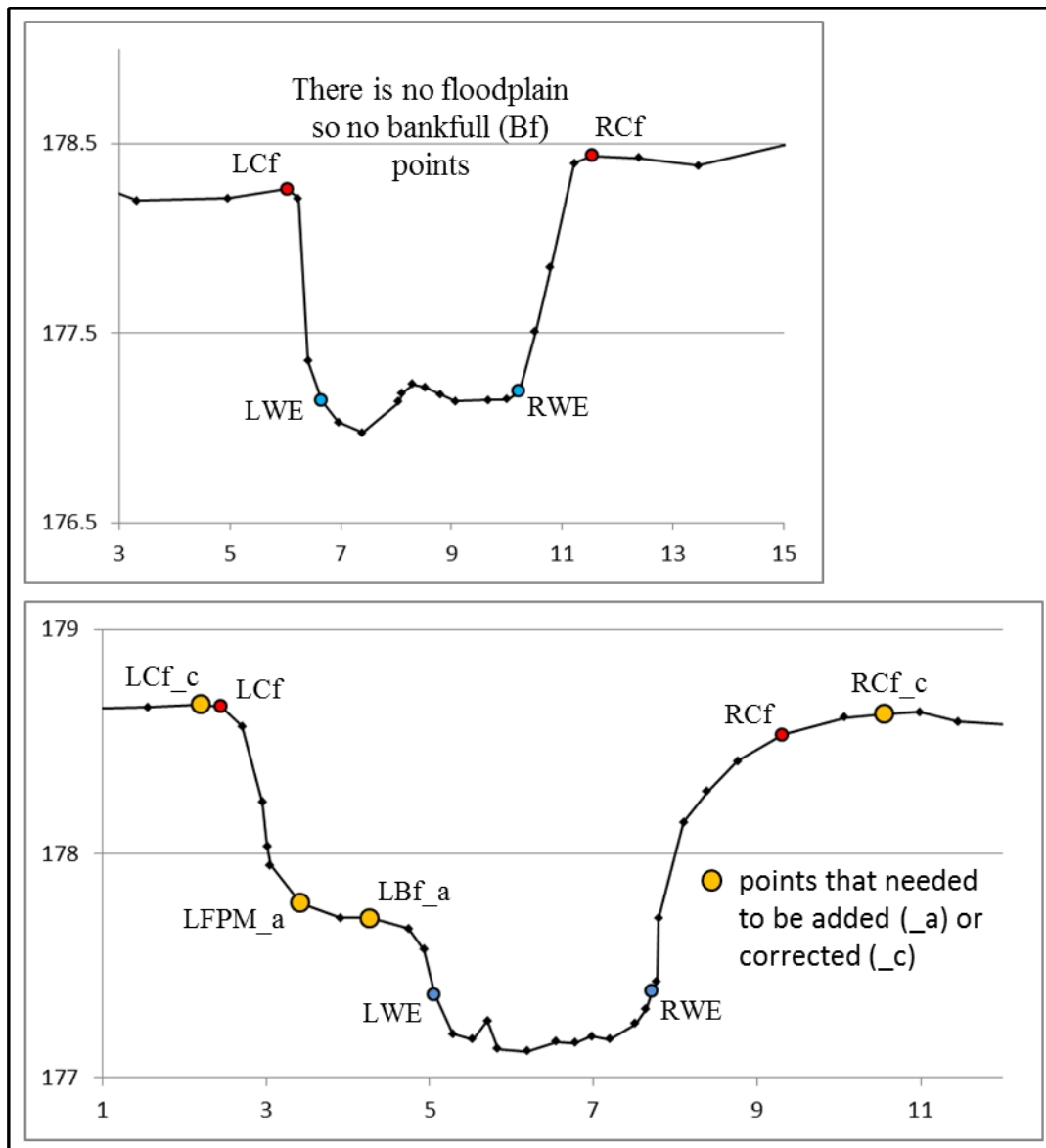


Figure 5. Examples of how transects should be plotted so that the location of the diagnostic points can be verified. Make sure that channel-full and bankfull points are placed on a flat surface (this can be difficult to accurately identify in the field so points may need to be corrected). Additionally, make sure that any points that should have been labeled are relabeled with the proper code (e.g., the left bankfull bank top point was labeled incorrectly as LOC so it was re-labeled LBf). Note: Do not add the _a or _c to any of the points as these suffixes were for illustrative purposes only.

	A	B	C	D	E	F	G	H	I
1	OBJECTID	Name	Code_raw	Code	Northing	Easting	Elevation	d2b	X5_dist
2	3	2001	LBMOC	LBMOC	3647807	617387.9	179.8099	7.619304	-7.6193
3	4	2002	LBMOC	LBMOC	3647807	617388.4	179.5092	7.059414	-7.05941
4	5	2003	LBMOC	LBMOC	3647807	617388.7	179.3822	6.846025	-6.84603
5	6	2004	LBMOC	LBMOC	3647807	617388.8	179.1975	6.655901	-6.6559
6	7	2005	LBMOC	LBMOC	3647807	617389.1	179.018	6.335442	-6.33544
7	8	2006	LBMOC	LBMOC	3647807	617389.4	178.9025	6.060297	-6.0603
8	9	2007	LBMOC	LBMOC	3647807	617389.6	178.7325	5.838343	-5.83834
9	10	2008	LBMOC	LBMOC	3647807	617389.9	178.5945	5.568252	-5.56825
10	11	2009	LBMOC	LBMOC	3647807	617390.1	178.5501	5.369831	-5.36983
11	12	2010	LBMOC	LBMOC	3647806	617390.5	178.4257	4.977638	-4.97764
12	13	2011	LBMOC	LBMOC	3647806	617390.9	178.4174	4.521611	-4.52161
13	14	2012	LBMOC	LBMOC	3647806	617391.3	178.423	4.151469	-4.15147
14	15	2013	LBMOC	LBMOC	3647806	617391.9	178.4208	3.556888	-3.55689
15	16	2014	LBMOC	LBMOC	3647807	617392.3	178.4175	3.189924	-3.18992
16	17	2015	LBMOC	LBMOC	3647806	617392.8	178.4491	2.721098	-2.7211
17	18	2016	LBMOC	LBMOC	3647806	617393.2	178.4533	2.213949	-2.21395
18	19	2017	LBMOC	LBMOC	3647806	617393.8	178.4725	1.689608	-1.68961
19	20	2018	LBMOC	LBMOC	3647806	617394.2	178.5243	1.194828	-1.19483
20	21	2019	LBMOC	LBMOC	3647806	617394.6	178.5946	0.808428	-0.80843
21	22	2020	LBMOC	LBMOC	3647806	617395	178.633	0.469098	-0.4691
22	1	BM2	BM	LBM	3647806	617395.4	178.6738	0	0
23	23	2021	LOC	LOC	3647806	617395.9	178.6568	0.527127	0.527127
24	24	2022	LOC	LOC	3647806	617396.4	178.6467	0.992413	0.992413
25	25	2023	LOC	LOC	3647806	617397	178.653	1.561747	1.561747
26	26	2024	LOC	LCf	3647806	617397.6	178.666	2.192124	2.192124
27	27	2025	LCf	LOC	3647806	617397.9	178.6538	2.454539	2.454539
28	28	2026	LOC	LOC	3647806	617398.1	178.5665	2.704581	2.704581

Figure 6. Copy and paste the Name, Code, Northing, Easting, and Elevation of each transect into its own Excel workbook and save as a .csv so that the data can be processed in R and cross-sectional metrics can be calculated.

	A	B	C	D	E	F
1	OBJECTID	Name	Code	Northing	Easting	Elevation
2	151	WS11	LONG	3647781	617403.7	177.709
3	72	WS10	LONG	3647785	617403	177.6001
4	71	WS09	LONG	3647791	617402.4	177.532
5	70	WS08	LONG	3647795	617402.1	177.5068
6	69	WS07	LONG	3647801	617401.7	177.4544
7	68	WS06	LONG	3647805	617402.1	177.4211
8	67	WS05	LONG	3647810	617401.1	177.4111
9	66	WS04	LONG	3647815	617400.6	177.409
10	65	WS03	LONG	3647820	617402.8	177.4001
11	64	WS02	LONG	3647825	617405	177.3333
12	63	WS01	LONG	3647830	617406.3	177.2166

Figure 7. Format the data so that the most upstream point (WS11) is first and the most downstream point (WS01) is last.

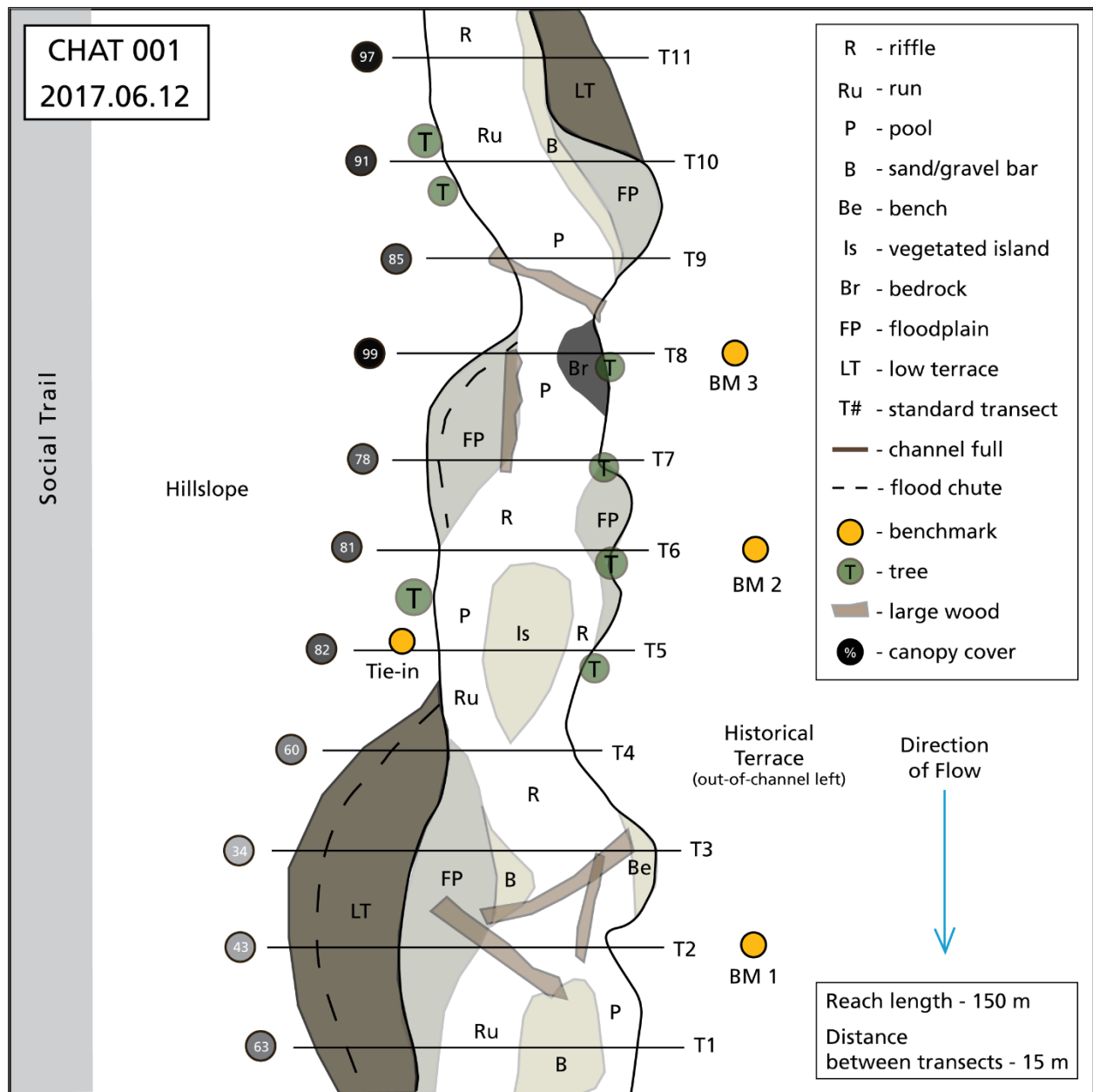


Figure 8. An example reach map. This illustration depicts the stream reach surveyed at CHAT 001.

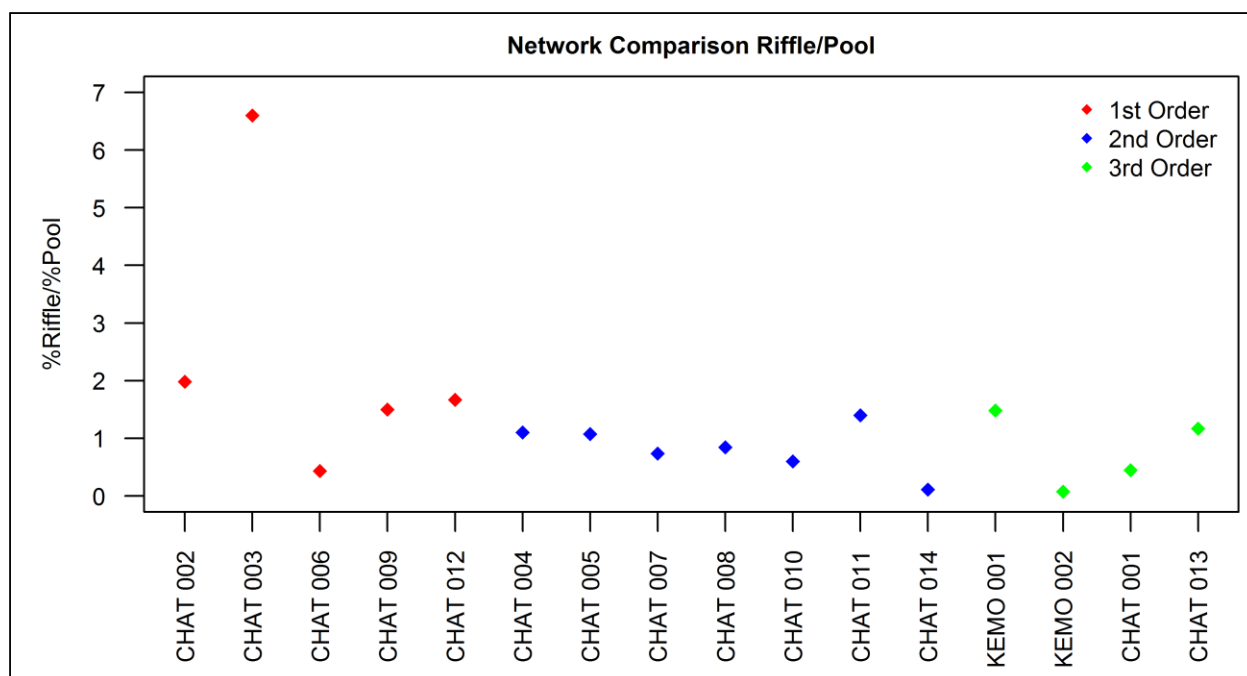


Figure 9. Example of a network comparison of the percent of each reach covered by riffle habitat divided by the percent of each reach covered by pool habitat. The sites are colored based on the Strahler stream order of the segment that the reach occupies.

Literature Cited

- Crawley, M. J. 2005. Statistics: an introduction using R. John Wiley & Sons, West Sussex, England.
- Fitzpatrick, F. A., I. R. Waite, P. J. D'Arconte, M. R. Meador, M. A. Maupin, and M. E. Gurtz. 1998. Revised methods for characterizing stream habitat in the National Water Quality Assessment Program: U.S. Geological Survey Water Resources Investigations Report 98-4052. Available at: <http://water.usgs.gov/nawqa/protocols/WRI98-4052/index.html> (last accessed December 2016).
- Helsel, D.R. and R. M. Hirsch. 2002. Statistical Methods in Water Resources Techniques of Water Resources Investigations, Book 4, chapter A3. U.S. Geological Survey.
- McDonald, J. M., M. B. Gregory, J. W. Riley, and E. N. Starkey. 2018a. Monitoring wadeable stream habitat conditions in southeast coast network parks: Protocol narrative. Natural Resource Report NPS/SECN/NRR—2018/1715. National Park Service, Fort Collins, Colorado. Available at: <https://irma.nps.gov/DataStore/Reference/Profile/2254874> (last accessed September 2018).
- McDonald, J.M., E. N. Starkey, and M. B. Gregory. 2018b. Wadeable stream reporting guide—Version 1.0. Southeast Coast Network Standard Operating Procedure NPS/SECN/SOP—2.2.15. National Park Service, Athens, Georgia. Available at: <https://irma.nps.gov/DataStore/Reference/Profile/2254020> (last accessed July 2018).
- McDonald, J. M., E. N. Starkey, P. Capece, and M. B. Gregory. 2018c. Wadeable stream data entry and storage—Version 1.0. Southeast Coast Network Standard Operating Procedure NPS/SECN/SOP—2.2.13. National Park Service, Athens, Georgia. Available at: <https://irma.nps.gov/DataStore/Reference/Profile/2254018> (last accessed July 2018).
- McDonald, J. M., M. B. Gregory, and J. Riley. 2018d. Wadeable stream basin- and segment-scale data summary—Version 1.0. Southeast Coast Network Standard Operating Procedure NPS/SECN/SOP—1.2.15. National Park Service, Athens, Georgia. Available at: <https://irma.nps.gov/DataStore/Reference/Profile/2254012> (last accessed July 2018).

Appendix A. R Code to Create LULC Table

```
# CHANGE working directory
setwd("S:/NEW DIRECTORY
STRUCTURE/Monitoring/StreamHabitat/Wadeable/Data/PARK/A_RawData")

# CHANGE site
site <- "PARK_###"

# CHANGE survey.date to the day of survey
survey.date <- "YYYY_MM_DD"

# NLCD class values
n.class <- c(11, 12, 21, 22, 23, 24, 31, 41, 42, 43, 52, 71, 81,
82, 90, 95)

# Name of NLCD classes
class.n <- c(
"Open Water",
"Perennial Snow/Ice",
"Dev-Open",
"Dev-Low",
"Dev-Med",
"Dev-High",
"Barren",
"Deciduous",
"Evergreen",
"Mixed Forest",
"Shrub",
"Grassland",
"Pasture",
"Cultivated",
"Woody Wetland",
"Herbaceous Wetland")

# Import the LULC tables
# To add additional years simply copy the lines from a previous
year and add new year
table.2001 <- read.table(paste(site, "_LULC_2001.txt", sep = ""),
header = TRUE, sep = ",")
table.2006 <- read.table(paste(site, "_LULC_2006.txt", sep = ""),
header = TRUE, sep = ",")
table.2011 <- read.table(paste(site, "_LULC_2011.txt", sep = ""),
header = TRUE, sep = ",")

table.2001[, 'Class'] <- table.2001[, 'OBJECTID'] - 1      # Adds a
field with correct LULC numbers
table.2006[, 'Class'] <- table.2006[, 'OBJECTID'] - 1
table.2011[, 'Class'] <- table.2011[, 'OBJECTID'] - 1
```

```

t.2001 <- table.2001[which(table.2001[, 'Class'] %in% n.class),]
# Gets rid of additional fields
t.2006 <- table.2006[which(table.2006[, 'Class'] %in% n.class),]
t.2011 <- table.2011[which(table.2011[, 'Class'] %in% n.class),]

t.2001[, 'Percent'] <- (t.2001[, 'Id_1']/sum(t.2001[, 'Id_1']))*100
# Calculates percent
t.2006[, 'Percent'] <- (t.2006[, 'Id_1']/sum(t.2006[, 'Id_1']))*100
t.2011[, 'Percent'] <- (t.2011[, 'Id_1']/sum(t.2011[, 'Id_1']))*100

# Combine the years and label classes
output <- data.frame(
  Class = class.n,
  "2001" = round(t.2001[, 'Percent'], 3),
  "2006" = round(t.2006[, 'Percent'], 3),
  "2011" = round(t.2011[, 'Percent'], 3))

# Write combined table to csv
write.csv(output, paste(site, "_LULC_", survey.date, ".csv", sep
=""))

```

Appendix B. R Code to Extract Cross-Sectional Data

```
# Extract Cross-Sectional Information from survey data

# Identify which PARK, Site, and Day of Survey
park <- "PARK"
site <- "###"
day <- "YYYYMMDD"

side <- ("L", "L", "L")

# Input reach median bankfull height (from standard transects)
# this will be used to calculate bankfull dimensions
# for cross-sections that do not have a floodplain
aBf.depth <- 0.45

# Set working directory
setwd(paste("S:/NEW DIRECTORY
STRUCTURE/Monitoring/StreamHabitat/Wadeable/Data/", park,
"/C_Certified/", park, site, sep = ""))

# Read in the transect data
t1 <- read.csv(paste(park, site, "_", day, "_T1.csv", sep = ""))
# Transect 1
t2 <- read.csv(paste(park, site, "_", day, "_T2.csv", sep = ""))
# Transect 2
t3 <- read.csv(paste(park, site, "_", day, "_T3.csv", sep = ""))
# Transect 3

# Function to calculate distance to benchmark
d2bm <- function(transect)
{
# Determine the Northing, Easting, and Elevation of the river left
benchmark
bm <- data.frame(
Northing = transect[which(transect[, 'Code'] == "LBM"), 'Northing'],
Easting = transect[which(transect[, 'Code'] == "LBM"), 'Easting'],
Elevation = transect[which(transect[, 'Code'] == "LBM"), 'Elevation'])

# Calculate the distance to the river left benchmark for the points
to the left of the benchmark
LoBM <- transect[which(transect[, 'Position'] == "LBMOC"), ]
LoBM[, 'd2BM'] <- -sqrt((LoBM[, 'Northing'] -
bm[, 'Northing'])^2 + (LoBM[, 'Easting'] - bm[, 'Easting'])^2)

# Calculate the distance to the river left benchmark for the points
to the right of the benchmark
RoBM <- transect[which(transect[, 'Position'] != "LBMOC"), ]
```

```

RoBM[, 'd2BM'] <- sqrt((RoBM[, 'Northing']-
bm[, 'Northing'])^2+(RoBM[, 'Easting']-bm[, 'Easting'])^2)

# Combine the points
rbind(LoBM, RoBM)
}

# Add a column with distance to benchmark
transect.1 <- d2bm(t1)
transect.2 <- d2bm(t2)
transect.3 <- d2bm(t3)

# Remove 'Position' column
transect.1$Position <- NULL
transect.2$Position <- NULL
transect.3$Position <- NULL

# Convert d2BM and elevation to Distances (from the furthest left
point) and Heights (above thalweg)
transect.1$Distance <- abs(min(transect.1$d2BM))+transect.1$d2BM
transect.1$Height <- transect.1$Elevation -
min(transect.1$Elevation)

transect.2$Distance <- abs(min(transect.2$d2BM))+transect.2$d2BM
transect.2$Height <- transect.2$Elevation -
min(transect.2$Elevation)

transect.3$Distance <- abs(min(transect.3$d2BM))+transect.3$d2BM
transect.3$Height <- transect.3$Elevation -
min(transect.3$Elevation)

# Determine thalweg elevation and
# the position and elevation of the
# Channel-full and Bankfull bank-tops
bank.position <- function(transect)
{
thalweg.y <- min(transect[, 'Height'])

LCf.x <- transect[which(transect[, 'Code']=="LCF"), 'Distance']
LCf.y <- transect[which(transect[, 'Code']=="LCF"), 'Height']

RCf.x <- transect[which(transect[, 'Code']=="RCF"), 'Distance']
RCf.y <- transect[which(transect[, 'Code']=="RCF"), 'Height']

LBf.x <- transect[which(transect[, 'Code']=="LBF"), 'Distance']
LBf.y <- transect[which(transect[, 'Code']=="LBF"), 'Height']

RBf.x <- transect[which(transect[, 'Code']=="RBF"), 'Distance']
RBf.y <- transect[which(transect[, 'Code']=="RBF"), 'Height']

```

```

data.frame(
  thalweg = thalweg.y,
  LCfx = LCf.x,
  LCfy = LCf.y,
  RCfx = RCf.x,
  RCfy = RCf.y,
  LBfx = if(length(LBf.x) == 0){9999} else{LBf.x},
  LBfy = if(length(LBf.y) == 0){9999} else{LBf.y},
  RBfx = if(length(RBf.x) == 0){9999} else{RBf.x},
  RBfy = if(length(RBf.y) == 0){9999} else{RBf.y})
}

t1.bp <- bank.position(transect.1)
t2.bp <- bank.position(transect.2)
t3.bp <- bank.position(transect.3)

# Subset transect points based on
# elevation (below minimum Cf elevation) &
# d2BM (right of LCf & left of RCf)

Cf.XS <- function(transect, bp)
{
  Cf.channel <- subset(transect, transect[, 'Height'] <=
    min(bp[, c('LCfy', 'RCfy')]) &
    transect[, 'Distance'] >=
    bp[, 'LCfx'] &
    transect[, 'Distance'] <=
    bp[, 'RCfx'])

  # Check to see if the left-most point Height is equal to Cf Height
  # if it is not, then an unknown point needs to be determined

  Cf.channel <- if(Cf.channel[1, 'Height'] <
    max(Cf.channel[, 'Height']))
  {
    # Calculate the (x,y) of last (potentially unknown) point
    # Determine Channel-full Height and x,y of right-most point on Cf
    XS
    Cf.y <- max(Cf.channel[, 'Height']) # Channel-full Height
    CfL.x <- Cf.channel[1, 'Distance']
    CfL.y <- Cf.channel[1, 'Height']

    # Extract the x,y of the point that is left of the left most point
    (LoL)
    LoL <- transect[which(transect[, 'Distance'] == CfL.x) - 1, ]

    # Difference in Height to unknown point

```

```

D2U.y <- LoL[, 'Height'] - Cf.y # LoL Height minus Cf Height

# Difference in Height to left-most point
D2L.y <- LoL[, 'Height'] - CfL.y

# Distance between LoL and left-most point
D2L.x <- abs(LoL[, 'Distance'] - CfL.x)

# Distance between RoR and unknown point
D2U.x <- D2U.y*tan(atan(D2L.x/D2L.y))

# Distance for unknown point
Lx.u <- LoL[, 'Distance'] + D2U.x

unL <- data.frame(
  Name = 9999,
  Code = "unL",
  Northing = 9999,
  Easting = 9999,
  Elevation = 9999,
  d2BM = 9999,
  Distance = Lx.u,
  Height = Cf.y)

rbind(unL, Cf.channel)
} else {Cf.channel}
# Check to see if the right-most point Height is equal to Cf Height
# if it is not, then an unknown point needs to be determined
Cf.channel <- if(Cf.channel[nrow(Cf.channel), 'Height'] <
max(Cf.channel[, 'Height']))
{
# Calculate the (x,y) of last (potentially unknown) point
# Determine Channel-full Height and x,y of right-most point on Cf
XS
Cf.y <- max(Cf.channel[, 'Height']) # Channel-full Height
CfR.x <- Cf.channel[nrow(Cf.channel), 'Distance']
CfR.y <- Cf.channel[nrow(Cf.channel), 'Height']

# Extract the x,y of the point that is right of the right most
point (RoR)
RoR <- transect[which(transect[, 'Distance']==CfR.x)+1,]

# Difference in Height to unknown point
D2U.y <- RoR[, 'Height'] - Cf.y # RoR Height minus Cf Height

# Difference in Height to right-most point
D2R.y <- RoR[, 'Height'] - CfR.y

# Distance between RoR and right-most point

```

```

D2R.x <- abs(RoR[, 'Distance'] - CfR.x)

# Distance between RoR and unknown point
D2U.x <- D2U.y*tan(atan(D2R.x/D2R.y))

# Distance for unknown point
Rx.u <- RoR[, 'Distance'] - D2U.x

unR <- data.frame(
  Name = 9999,
  Code = "unR",
  Northing = 9999,
  Easting = 9999,
  Elevation = 9999,
  d2BM = 9999,
  Distance = Rx.u,
  Height = Cf.y)

rbind(Cf.channel, unR)
} else{Cf.channel}
}

t1.Cf.xs <- Cf.XS(transect.1, t1.bp)
t2.Cf.xs <- Cf.XS(transect.2, t2.bp)
t3.Cf.xs <- Cf.XS(transect.3, t3.bp)

# Extract bankfull XS points
Bf.XS <- function(transect, bp)
{
  Bf.y <- if(min(bp[,c('LBfy', 'RBfy')])<9999){
    min(bp[,c('LBfy', 'RBfy')])
  } else{bp[, 'thalweg'] + aBf.depth}

  Bf.channel <- subset(transect, transect[, 'Height'] <= Bf.y &
    transect[, 'Distance'] >
    bp[, 'LCfx'] &
    transect[, 'Distance'] <
    bp[, 'RCfx'])

# Check to see if the left-most point Height is equal to Bf Height
# if it is not, then an unknown point needs to be determined

Bf.channel <- if(Bf.channel[1, 'Height'] < Bf.y){
  # Calculate the (x,y) of last (potentially unknown) point
  # Determine x,y of right-most point on Bf XS
  BfL.x <- Bf.channel[1, 'Distance']
  BfL.y <- Bf.channel[1, 'Height']

```

```

# Extract the x,y of the point that is left of the left most point
(LoL)
LoL <- transect[which(transect[, 'Distance'] == BfL.x) - 1,]

# Difference in Height to unknown point
D2U.y <- LoL[, 'Height'] - Bf.y # LoL Height minus Bf Height

# Difference in Height to left-most point
D2L.y <- LoL[, 'Height'] - BfL.y

# Distance between LoL and left-most point
D2L.x <- abs(LoL[, 'Distance'] - BfL.x)

# Distance between RoR and unknown point
D2U.x <- D2U.y * tan(atan(D2L.x / D2L.y))

# Distance for unknown point
Lx.u <- LoL[, 'Distance'] + D2U.x

unL <- data.frame(
  Name = 9999,
  Code = "unL",
  Northing = 9999,
  Easting = 9999,
  Elevation = 9999,
  d2BM = 9999,
  Distance = Lx.u,
  Height = Bf.y)

rbind(unL, Bf.channel)
} else {Bf.channel}

# Check to see if the right-most point Height is equal to Bf Height
# if it is not, then an unknown point needs to be determined
Bf.channel <- if(Bf.channel[nrow(Bf.channel), 'Height'] < Bf.y)
{
  # Calculate the (x,y) of last (potentially unknown) point
  # Determine x,y of right-most point on Bf XS
  BfR.x <- Bf.channel[nrow(Bf.channel), 'Distance']
  BfR.y <- Bf.channel[nrow(Bf.channel), 'Height']

  # Extract the x,y of the point that is right of the right most
  point (RoR)
  RoR <- transect[which(transect[, 'Distance'] == BfR.x) + 1,]

  # Difference in Height to unknown point
  D2U.y <- RoR[, 'Height'] - Bf.y # RoR Height minus Bf Height

  # Difference in Height to right-most point

```

```

D2R.y <- RoR[, 'Height'] - BfR.y

# Distance between RoR and right-most point
D2R.x <- abs(RoR[, 'Distance'] - BfR.x)

# Distance between RoR and unknown point
D2U.x <- D2U.y*tan(atan(D2R.x/D2R.y))

# Distance for unknown point
Rx.u <- RoR[, 'Distance'] - D2U.x

unR <- data.frame(
  Name = 9999,
  Code = "unR",
  Northing = 9999,
  Easting = 9999,
  Elevation = 9999,
  d2BM = 9999,
  Distance = Rx.u,
  Height = Bf.y)

rbind(Bf.channel, unR)
} else{Bf.channel}
}

t1.Bf.xs <- Bf.XS(transect.1, t1.bp)
t2.Bf.xs <- Bf.XS(transect.2, t2.bp)
t3.Bf.xs <- Bf.XS(transect.3, t3.bp)


#####
##### CALCULATE XS AREA, WETTED PERIMETER, and HYDRAULIC RADIUS
#####
###
##
#

#####
### BE SURE TO CHECK CF and BF transects so no extra points were
included!!!!

# FUNCTION - CALCULATE AREAS and PERIMETER
xs.calculate <- function(xs)
{
n <- nrow(xs)-1

a.1 <- numeric(n)
a.2 <- numeric(n)

```

```

perimeter <- numeric(n)
depth <- numeric(n)

for (i in 1:n)
{
if(xs[i,'Height'] > xs[i+1,'Height'])
{
a.1[i] <- (max(xs[, 'Height']) - xs[i,'Height']) *
(abs(xs[i,'Distance']-xs[i+1,'Distance']))
a.2[i] <- (abs(xs[i,'Height']-xs[i+1,'Height']) *
(abs(xs[i,'Distance'] - xs[i+1,'Distance'])))/2
perimeter[i] <- sqrt(abs(xs[i,'Height']-xs[i+1,'Height'])^2 +
abs(xs[i,'Distance'] - xs[i+1,'Distance'])^2)
#depth[i] <- max(xs[, 'Height']) - xs[i,'Height']
} else{a.1[i] <- (max(xs[, 'Height']) - xs[i+1,'Height']) *
(abs(xs[i,'Distance']-xs[i+1,'Distance']))
a.2[i] <- (abs(xs[i,'Height']-xs[i+1,'Height']) *
(abs(xs[i,'Distance'] - xs[i+1,'Distance'])))/2
perimeter[i] <- sqrt(abs(xs[i,'Height']-xs[i+1,'Height'])^2
+ abs(xs[i,'Distance'] - xs[i+1,'Distance'])^2)
#depth[i] <- max(xs[, 'Height']) - xs[i,'Height']
}}

xs.area <- sum(a.1) + sum(a.2)
xs.perimeter <- sum(perimeter)
xs.hydraulic.radius <- xs.area/xs.perimeter
xs.width <- max(xs[, 'Distance']) - min(xs[, 'Distance'])
xs.depth <- max(xs[, 'Height']) - min(xs[, 'Height'])
#xs.depth.avg <- sum(depth)/n
xs.wd.ratio <- xs.width/xs.depth

data.frame(
Area = xs.area,
Perimeter = xs.perimeter,
Hydraulic_Radius = xs.hydraulic.radius,
Width = xs.width,
Depth = xs.depth,
#Avg_Depth = xs.depth.avg,
#Area_Avg = xs.width * xs.depth.avg,
WD_ratio = xs.wd.ratio)
}

# Function to calculate XS metrics and combine
xs.combined <- function(bf, cf)
{
Bf <- xs.calculate(bf)
Cf <- xs.calculate(cf)
ratio <- Cf/Bf

```

```

output <- rbind(Bf, Cf, ratio)
rownames(output) <- c("Bankfull", "Channel-full", "Ratio")
return(output)
}

write.csv(xs.combined(t1.Bf.xs,t1.Cf.xs),paste(park, site, day,
"T1_calculated.csv", sep = "_"))
write.csv(xs.combined(t2.Bf.xs,t2.Cf.xs),paste(park, site, day,
"T2_calculated.csv", sep = "_"))
write.csv(xs.combined(t3.Bf.xs,t3.Cf.xs),paste(park, site, day,
"T3_calculated.csv", sep = "_"))

#####
##### PLOT Individual Cross-Sections

# Determine axis' range
x.max <- max(c(transect.1$Distance, transect.2$Distance,
transect.3$Distance))
y.max <- max(c(transect.1$Height, transect.2$Height,
transect.3$Height))

# Specify axis labels (Will be dependent on maxima)
x.axis <- seq(0,30,10)
x.labels <- c(0,10,20,30)

y.axis <- seq(0,3,1)
y.labels <- c(0,1,2,3)

# Create objects with diagnostic points
pts4plots <- function(transect, Cf, Bf)
{
data.frame(
LBMx = transect[which(transect[, 'Code'] == "LBM"), 'Distance'],
LBMy = transect[which(transect[, 'Code'] == "LBM"), 'Height'],
RBMx = transect[which(transect[, 'Code'] == "RBM"), 'Distance'],
RBMy = transect[which(transect[, 'Code'] == "RBM"), 'Height'],
thalx = Cf[which(Cf[, 'Height'] == 0), 'Distance'],
thaly = 0,
LCfx = transect[which(transect[, 'Code'] == "LCF"), 'Distance'],
LCfy = transect[which(transect[, 'Code'] == "LCF"), 'Height'],
RCfx = transect[which(transect[, 'Code'] == "RCF"), 'Distance'],
RCfy = transect[which(transect[, 'Code'] == "RCF"), 'Height'],
LBfx = if(Bf[1, 'Code'] == "unL"){Bf[1, 'Distance']}
} else{transect[which(transect[, 'Code'] == "LBF"), 'Distance']},
LBfy = if(Bf[1, 'Code'] == "unL"){Bf[1, 'Height']}
} else{transect[which(transect[, 'Code'] == "LBF"), 'Height']},
RBfx = if(Bf[nrow(Bf), 'Code'] == "unR"){Bf[nrow(Bf), 'Distance']}
} else{transect[which(transect[, 'Code'] == "RBF"), 'Distance']},
RBfy = if(Bf[nrow(Bf), 'Code'] == "unR"){Bf[nrow(Bf), 'Height']}

```

```

} else{transect[which(transect[, 'Code'] == "RBF"), 'Height' ]}}
}

t1.p4p <- pts4plots(transect.1, t1.Cf.xs, t1.Bf.xs)
t2.p4p <- pts4plots(transect.2, t2.Cf.xs, t2.Bf.xs)
t3.p4p <- pts4plots(transect.3, t3.Cf.xs, t3.Bf.xs)

# PLOT Detailed Transect 1
png(filename = paste(park, site, day, "T1_plot.png", sep = "_"),
units = "in",
width = 6.5,
height = 3.5,
res = 600)

par(mai=c(0.6,0.64,0.33,0.1))

plot(
transect.1[, 'Distance'],
transect.1[, 'Height'],
pch = 20,
cex = 0.75,
cex.lab = 0.75,
xlim = c(0, x.max),
ylim = c(0, y.max),
xaxt = "n",
yaxt = "n",
ann = FALSE)
lines(transect.1[, 'Distance'], transect.1[, 'Height'])

axis(1, at = x.axis, labels = x.labels, cex.axis = 0.75)
mtext("Distance (m)", 1, line = 1.5, cex = 0.75)

axis(2, at = y.axis, labels = y.labels, las=2, cex.axis = 0.75)
mtext("Height (m)", 2, line = 1.75, cex = 0.75)

x.Cf <-
c(t1.Cf.xs[, 'Distance'], sort(t1.Cf.xs[, 'Distance'], decreasing =
TRUE))
y.Cf <-
c(t1.Cf.xs[, 'Height'], rep(max(t1.Cf.xs[, 'Height']), nrow(t1.Cf.xs)))
polygon(x.Cf, y.Cf, col= rgb(0,0,255,50,maxColorValue=255))

x.Bf <-
c(t1.Bf.xs[, 'Distance'], sort(t1.Bf.xs[, 'Distance'], decreasing =
TRUE))
y.Bf <-
c(t1.Bf.xs[, 'Height'], rep(max(t1.Bf.xs[, 'Height']), nrow(t1.Bf.xs)))
polygon(x.Bf, y.Bf, col= rgb(0,0,255,100,maxColorValue=255))

```

```

points(t1.p4p[, 'LBMx'], t1.p4p[, 'LBMy'], pch = 24, bg = "RED")
points(t1.p4p[, 'RBMx'], t1.p4p[, 'RBMy'], pch = 24, bg = "RED")
points(t1.p4p[, 'thalx'], t1.p4p[, 'thaly'], pch = 21, bg =
"SKYBLUE")
points(t1.p4p[, 'LCfx'], t1.p4p[, 'LCfy'], pch = 21, bg = "ORANGE")
points(t1.p4p[, 'RCfx'], t1.p4p[, 'RCfy'], pch = 21, bg = "ORANGE")
points(t1.p4p[, 'LBfx'], t1.p4p[, 'LBfy'], pch = 21, bg = "GREEN")
points(t1.p4p[, 'RBfx'], t1.p4p[, 'RBfy'], pch = 21, bg = "GREEN")

title(paste(park, site, "- Detailed Transect 1", sep = " "), cex.main
= 0.75)

legend("bottomright",
inset = 0.01,
c("Benchmark", "Thalweg", "Channel-full", "Bankfull"),
pch = c(24, 21, 21, 21),
pt.bg = c("RED", "SKYBLUE", "ORANGE", "GREEN"),
cex = 0.75,
bty = "n")

box(which = "outer")

dev.off()

# PLOT Detailed Transect 2
png(filename = paste(park, site, day, "T2_plot.png", sep = "_"),
units = "in",
width = 6.5,
height = 3.5,
res = 600)

par(mai=c(0.6,0.64,0.33,0.1))

plot(
transect.2[, 'Distance'],
transect.2[, 'Height'],
pch = 20,
cex = 0.75,
cex.lab = 0.75,
xlim = c(0, x.max),
ylim = c(0, y.max),
xaxt = "n",
yaxt = "n",
ann = FALSE)
lines(transect.2[, 'Distance'], transect.2[, 'Height'])

axis(1, at = x.axis, labels = x.labels, cex.axis = 0.75)
mtext("Distance (m)", 1, line = 1.5, cex = 0.75)

```

```

axis(2, at = y.axis, labels = y.labels, las=2, cex.axis = 0.75)
mtext("Height (m)", 2, line = 1.75, cex = 0.75)

x.Cf <-
c(t2.Cf.xs[, 'Distance'], sort(t2.Cf.xs[, 'Distance'], decreasing =
TRUE))
y.Cf <-
c(t2.Cf.xs[, 'Height'], rep(max(t2.Cf.xs[, 'Height']), nrow(t2.Cf.xs)))
polygon(x.Cf, y.Cf, col= rgb(0,0,255,50,maxColorValue=255))

x.Bf <-
c(t2.Bf.xs[, 'Distance'], sort(t2.Bf.xs[, 'Distance'], decreasing =
TRUE))
y.Bf <-
c(t2.Bf.xs[, 'Height'], rep(max(t2.Bf.xs[, 'Height']), nrow(t2.Bf.xs)))
polygon(x.Bf, y.Bf, col= rgb(0,0,255,100,maxColorValue=255))

points(t2.p4p[, 'LBMx'], t2.p4p[, 'LBMy'], pch = 24, bg = "RED")
points(t2.p4p[, 'RBMx'], t2.p4p[, 'RBMy'], pch = 24, bg = "RED")
points(t2.p4p[, 'thalx'], t2.p4p[, 'thaly'], pch = 21, bg =
"SKYBLUE")
points(t2.p4p[, 'LCfx'], t2.p4p[, 'LCfy'], pch = 21, bg = "ORANGE")
points(t2.p4p[, 'RCfx'], t2.p4p[, 'RCfy'], pch = 21, bg = "ORANGE")
points(t2.p4p[, 'LBfx'], t2.p4p[, 'LBfy'], pch = 21, bg = "GREEN")
points(t2.p4p[, 'RBfx'], t2.p4p[, 'RBfy'], pch = 21, bg = "GREEN")

title(paste(park, site, "- Detailed Transect 2", sep = " "), cex.main
= 0.75)

legend("bottomright",
inset = 0.01,
c("Benchmark", "Thalweg", "Channel-full", "Bankfull"),
pch = c(24, 21, 21, 21),
pt.bg = c("RED", "SKYBLUE", "ORANGE", "GREEN"),
cex = 0.75,
bty = "n")

box(which = "outer")

dev.off()

# PLOT Detailed Transect 3
png(filename = paste(park, site, day, "T3_plot.png", sep = "_"),
units = "in",
width = 6.5,
height = 3.5,
res = 600)

```

```

par(mai=c(0.6,0.64,0.33,0.1))

plot(
  transect.3[, 'Distance'],
  transect.3[, 'Height'],
  pch = 20,
  cex = 0.75,
  cex.lab = 0.75,
  xlim = c(0, x.max),
  ylim = c(0, y.max),
  xaxt = "n",
  yaxt = "n",
  ann = FALSE)
lines(transect.3[, 'Distance'], transect.3[, 'Height'])

axis(1, at = x.axis, labels = x.labels, cex.axis = 0.75)
mtext("Distance (m)", 1, line = 1.5, cex = 0.75)

axis(2, at = y.axis, labels = y.labels, las=2, cex.axis = 0.75)
mtext("Height (m)", 2, line = 1.75, cex = 0.75)

x.Cf <-
c(t3.Cf.xs[, 'Distance'], sort(t3.Cf.xs[, 'Distance'], decreasing =
TRUE))
y.Cf <-
c(t3.Cf.xs[, 'Height'], rep(max(t3.Cf.xs[, 'Height']), nrow(t3.Cf.xs)))
polygon(x.Cf, y.Cf, col= rgb(0,0,255,50,maxColorValue=255))

x.Bf <-
c(t3.Bf.xs[, 'Distance'], sort(t3.Bf.xs[, 'Distance'], decreasing =
TRUE))
y.Bf <-
c(t3.Bf.xs[, 'Height'], rep(max(t3.Bf.xs[, 'Height']), nrow(t3.Bf.xs)))
polygon(x.Bf, y.Bf, col= rgb(0,0,255,100,maxColorValue=255))

points(t3.p4p[, 'LBMx'], t3.p4p[, 'LBMx'], pch = 24, bg = "RED")
points(t3.p4p[, 'RBMx'], t3.p4p[, 'RBMx'], pch = 24, bg = "RED")
points(t3.p4p[, 'thalx'], t3.p4p[, 'thalx'], pch = 21, bg =
"SKYBLUE")
points(t3.p4p[, 'LCfx'], t3.p4p[, 'LCfy'], pch = 21, bg = "ORANGE")
points(t3.p4p[, 'RCfx'], t3.p4p[, 'RCfy'], pch = 21, bg = "ORANGE")
points(t3.p4p[, 'LBfx'], t3.p4p[, 'LBfy'], pch = 21, bg = "GREEN")
points(t3.p4p[, 'RBfx'], t3.p4p[, 'RBfy'], pch = 21, bg = "GREEN")

title(paste(park, site, "- Detailed Transect 3", sep = " "), cex.main
= 0.75)

legend("bottomright",

```

```

inset = 0.01,
c("Benchmark", "Thalweg", "Channel-full", "Bankfull"),
pch = c(24, 21, 21, 21),
pt.bg = c("RED", "SKYBLUE", "ORANGE", "GREEN"),
cex = 0.75,
bty = "n")

box(which = "outer")

dev.off()

#####
##### PLOT Combined Cross-Sections

# Determine the max distance from origin for LCF
t1.lcf <- transect.1[which(transect.1[, 'Code'] ==
'LCF'), 'Distance']
t2.lcf <- transect.2[which(transect.2[, 'Code'] ==
'LCF'), 'Distance']
t3.lcf <- transect.3[which(transect.3[, 'Code'] ==
'LCF'), 'Distance']
lcf.max <- max(t1.lcf, t2.lcf, t3.lcf)

# Determine the LCF correction factor
t1.correction <- lcf.max - t1.lcf
t2.correction <- lcf.max - t2.lcf
t3.correction <- lcf.max - t3.lcf

# Standardize the LCF locations
transect.1[, 'new_Dist'] <- transect.1[, 'Distance'] + t1.correction
transect.2[, 'new_Dist'] <- transect.2[, 'Distance'] + t2.correction
transect.3[, 'new_Dist'] <- transect.3[, 'Distance'] + t3.correction

# Define axis' range (standardize across sites in report)
x.max <- 30
y.max <- 3

# Specify axis labels (Will be dependent on maxima)
x.axis <- seq(0, 30, 10)
x.labels <- c(0, 10, 20, 30)

y.axis <- seq(0, y.max, 1)
y.labels <- y.axis

# PLOT All Transects
png(filename = paste(park, site, "_", day, "_all_plot.png", sep =
""),
units = "in",

```

```

width = 6.5,
height = 3.5,
res = 600)

par(mai=c(0.6,0.64,0.33,0.1))

plot(
transect.1[, 'new_Dist'],
transect.1[, 'Height'],
type = "n",
cex = 0.75,
cex.lab = 0.75,
xlim = c(0, x.max),
ylim = c(0, y.max),
xaxt = "n",
yaxt = "n",
ann = FALSE)
lines(transect.1[, 'new_Dist'], transect.1[, 'Height'], col = 'red')

lines(transect.2[, 'new_Dist'], transect.2[, 'Height'], col = 'blue')

lines(transect.3[, 'new_Dist'], transect.3[, 'Height'], col =
'orange')

axis(1, at = x.axis, labels = x.labels, cex.axis = 0.75)
mtext("Distance (m)", 1, line = 1.5, cex = 0.75)

axis(2, at = y.axis, labels = y.labels, las=2, cex.axis = 0.75)
mtext("Height (m)", 2, line = 1.75, cex = 0.75)

title(paste(park, site, "- Detailed Transects", sep = " "), cex.main
= 0.75)

legend("bottomright",
inset = 0.01,
c("Transect 1", "Transect 2", "Transect 3"),
lty = c(1,1,1),
col = c('red', 'blue', 'orange'),
cex = 0.75,
bty = "n")

box(which = "outer")

dev.off()

#####
####  END  ####
#####

```

Appendix C. R Code to Plot Longitudinal Profile

```
# PLOT Longitudinal Profile

# Identify which PARK, Site, and Day of Survey
park <- "HOBE"
site <- "001"
day <- "2016_07_26"

# Set working directory
setwd(paste("S:/NEW DIRECTORY
STRUCTURE/Monitoring/StreamHabitat/Wadeable/Data/",
park,"/C_Certified", sep = ""))

# Read in the transect data
long <- read.csv(paste(park, site, day, "Long.csv", sep = "_"))
# Longitudinal Profile points

# Calculate distance between points
dbp <- c(rep(0,11))
for(i in 2:11)
{
dbp[i] <- sqrt((long[i,'Northing']-long[i-
1,'Northing'])^2+(long[i,'Easting']-long[i-1,'Easting'])^2)
}

long$DbP <- dbp

# Calculate Distance along profile
distance <- c(rep(0,11))
for(i in 2:11)
{
distance[i] <- long[i,'DbP']+distance[i-1]
}

long$Distance <- distance

# Determine Height of point above profile minimum elevation
height <- c(rep(0,11))
for(i in 1:11)
{
height[i] <- long[i,'Elevation']-min(long[, 'Elevation'])
}

long$Height <- height

# Determine profile trend
trend <- lm(long$Height ~ long$Distance)
r2.round <- round(summary(trend)$r.squared,2)
```

```

intercept <- coef(trend)[1]
slope <- coef(trend)[2]

equation <- paste("y = ", round(slope,3),"x +
",round(intercept,2),sep = "")
r2.label <- bquote(r^2 == .(r2.round))

# Specify axis labels (Will be dependent on maxima)
x.axis <- seq(0,50,10)
y.axis <- seq(0,0.5,0.25)

# PLOT Longitudinal Profile
png(filename = paste(park, site, day, "Longitudinal_Profile.png",
sep = "_"),
units = "in",
width = 6.5,
height = 3.5,
res = 600)

par(mai=c(0.6,0.64,0.33,0.1))

plot(
long[, 'Distance'],
long[, 'Height'],
pch = 20,
cex = 0.75,
cex.lab = 0.75,
xaxt = "n",
yaxt = "n",
ann = FALSE)

title(paste(park,site,"- Longitudinal Profile", sep = " "),
cex.main = 0.75)

axis(1, at = x.axis, labels = x.axis, cex.axis = 0.75)
mtext("Distance (m)", 1, line = 1.5, cex = 0.75)

axis(2, at = y.axis, labels = y.axis, las=0, cex.axis = 0.75)
mtext("Height (m)", 2, line = 1.9, cex = 0.75)

lines(long[, 'Distance'],long[, 'Height'])
abline(trend, col = "RED")

text(0,0.045,r2.label, pos = 4, cex = 0.75)
text(0,0.080,equation, pos = 4, cex = 0.75)

box(which = "outer")

dev.off()

```