Creation of a Geospatial Tool to Locate Optimal Reference Reach Sites for Natural Channel Design Restoration Methods

By
Clare Maher

A Master's paper submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the degree of
Master of Natural Resources

Hydrology

Raleigh, North Carolina

2019

Approved by advisory committee:

Dr. Joshua Gray, Chair
Dr. Laura Tateosian
Dr. Barbara Doll

Date of final exam:
June 27, 2019

ABSTRACT

MAHER, CLARE. Master of Natural Resources, Hydrology. *Creation of a Geospatial Tool to Locate Optimal Reference Reach Sites for Natural Channel Design Restoration Methods.*

In fluvial geomorphology, natural channel design techniques are often used for stream restoration in order to create stable channels and increase stream functions as much as possible. Establishing design criteria for a restoration project is one of the most critical components of a successful natural channel design. The most common method of establishing numerical criteria for channel design characteristics such as dimension, pattern and profile is through a reference reach survey. A reference reach is a stable channel segment that serves as a model for restoration, and is described by quantitative field data of its morphological features. Analyzing the suitability of a new reference reach site requires an extensive field survey to determine critical morphological metrics. Therefore, finding new reference reach sites is time consuming and often impractical. Additionally, it can be difficult to find new reference reaches due to widespread growth of urban, suburban and agricultural land use in the United States. Stream restoration designers often use the same reference reach sites for several projects due to data availability. Although many reference reaches can serve as valid stream design models for multiple projects, they might not be the best fit when considering all morphological characteristics of the stream. When working with a small selection of reference reach sites, it is more likely that multiple sites will need to be used in the design process to achieve the optimal ecological function. Given access to a wider selection of reference reach sites could yield higher quality restoration designs and streamline the design process. Ultimately, to adequately select a reference reach site for a degraded stream channel, ecological restoration designers must be knowledgeable of its fluvial geomorphology, which includes its natural function and interactions with the surrounding landscape.

The function and landscape interactions of streams are constantly changing, but an understanding of these morphological characteristics can aid in determining stream stability and response to stressors. Stressors that can reduce the stability of a stream system include climate change, land use/land cover change, and artificial hydrologic structures. These stressors all affect the streamflow and/or sediment load of a stream in some way, which ultimately changes its morphology. For example, urban development can decrease the floodplain width and cause channelization of a stream. The absence of a wide floodplain increases the amount and velocity of water flowing through a stream, which erodes the banks and alters the physical shape of the channel. Loss of stability in a stream system not only alters the physical characteristics, but also affects the overall ecological health of a stream. When morphological changes occur, a loss of biological diversity within the stream ecosystem can also occur due to habitat alteration or loss. Knowledge of the fluvial geomorphology and surrounding land use of a stream channel helps restoration professionals to predict its response to stressors and create a design plan that reflects that.

In natural channel design, the reference reach is a critical component of a restoration project's design plan. Restoration projects are often contracted from state agencies to private mitigation companies. Therefore, mitigation reports and related project data are made publically available. Although not required, reference reach locations and their morphological data are usually included within mitigation reports. A reference reach database was created using publically available data from the North Carolina Department of Mitigation Services (DMS). This database is used as the foundation for the Reference Reach Finder Tool (RRFT). The purpose of this project is to make the identification of reference reaches easier by creating a Geographic Information Systems (GIS)- based tool. We believe that this RRFT will improve the stream restoration design process and have important benefits to natural ecosystems.

ACKNOWLEDGEMENTS

I am grateful to all of those whom I have had the pleasure to work with during this project. This work would not have been possible without the support and guidance from my advisor, Dr. Joshua Gray, who gave me the opportunity to pursue my graduate studies at North Carolina State University. He has worked actively to help me to achieve my academic goals and prepare me for a rewarding career in the fields of natural resources and geospatial sciences. This project would also not have been possible without my internship experience at Resource Environmental Solutions, LLC, in Raleigh, NC. RES taught me about stream restoration design and the mitigation industry. During my internship, RES assigned me the reference reach database project, which ultimately led to the creation of this Master's project.

I am indebted to my committee member, Dr. Laura Tateosian, who patiently taught me how to write code in Python, which ultimately gave me the knowledge and skills to turn my rudimentary ideas into a successful project. I am thankful for all of the time she spent with me outside of class to assure that I would meet all of the project goals. Dr. Tateosian constantly challenged me to discover new methods and answers to coding problems, which tremendously improved my Python skills. Throughout some of the toughest parts of the project, she never seemed to doubt my competence to overcome them and move forward.

I would also like to acknowledge my committee member, Dr. Barbara Doll, who has helped me to understand some of the fundamental concepts of stream restoration. She has supported the value and validity of this Master's project from the moment when I first spoke with her and shared my ideas. Her enthusiasm for reference reach research instilled an initial confidence within me to pursue the project.

Personally, nobody has been more important to me in the pursuit of this project than my mother, Barbara. She was a constant source of encouragement, support and love throughout my graduate studies. My mom is the most courageous person I know, and she inspired me to keep moving forward each day during this process.

Finally, I am grateful for my friends, Michelle Lopez, Catherine Manner and Meredith Stull, whom I met during my time at NC State. Their positivity, laughter, encouragement, and confidence in my abilities allowed me to succeed.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1.0 Introduction

Over 3.5 million miles of streams and rivers flow through the United States, covering many diverse ecosystems (EPA 2019). In ecology, a stream is the general term used to describe all types of running water over a landscape. Size is usually the only distinction among the different terms that are commonly used to describe streams. For example, streams can also be referred to as rivers, creeks, or brooks, with the former as the largest type and the latter terms as smaller types (Richardson 2016).

In hydrology, the volume of streamflow per unit time is called discharge (ft^3/s), and hydrographs are used as a visual tool to analyze patterns of discharge. Hydrographs are useful when monitoring seasonal changes in streamflow, or when studying the impact of an extreme rainfall event, such as a hurricane. The United States Geological Survey installs stream gages in order to monitor discharge, and data is recorded in fifteen to sixty minute intervals (USGS 2019). Streamflow patterns are diverse and depend on several factors such as topography, climate, soil types and surrounding land use (Doll et al.).

The topography of a landscape influences the channel shape and flow of streams. Streams flow from areas of higher elevation to areas of lower elevation. A watershed is the land area that drains to a stream (Doll et al.). When precipitation falls on a watershed, it is evaporated back into the atmosphere, infiltrates the soil, or becomes surface runoff (Doll et al.). Surface runoff is also commonly referred to as overland flow, which travels downslope until it reaches a stream, lake, or other waterbody.

Watersheds contain a network of surface waterbodies that drain through a single outlet point. Therefore, the upstream health of the water and surrounding land area affects the health of downstream areas of a watershed. The watershed area determines the flow patterns and channel size of a stream. There are three main types of streams that are categorized by the presence of water flowing through a channel. Ephemeral streams are generally located at higher elevations within a watershed, and they only flow directly after a rainfall event (Doll et al.). Intermittent streams are located downstream from ephemeral streams, and they are characterized by seasonal streamflow during wet months (Doll et al.). Lastly, perennial streams have streamflow present year-round because they are located at low elevations within a watershed, and therefore receive the most drainage. Because of this, perennial streams also tend to have the largest channel size of the three types. The general size of stream channels can be described using the Strahler Stream Order classification method (Strahler 1957).

Stream orders are used as a hierarchical classification system, which refer to the relative size of streams. The smallest tributaries are called first-order streams, and the order increases at confluences- the point where two or more streams meet. For example, if two first-order headwater streams meet, the confluence is the origin of a second-order stream segment. The width of channels increase with each increasing stream order. For context, the Mississippi River is classified as a tenth-order river (Pierson et al. 2008).

Drainage basin, river basin and catchment are terms that are often used interchangeably with watershed. In the United States, the USGS delineates watersheds at various scales called Hydrologic Unit Codes (HUC). The United States is divided into four levels of successively smaller hydrologic units. These levels include regions, sub-regions, accounting units, and cataloging units which are identified by a unique HUC (USGS 2018). In recent years, more localized hydrologic unit classification has been developed to include a fifth-level category called watershed, and a sixth-level category called sub-watershed (NRCS 2007). The unique hydrologic unit codes range from two to twelve digits depending on the classification level. For example, 2-digit hydrologic unit codes are assigned to regions and 12-digit hydrologic unit codes are assigned to sub-watersheds (NRCS 2007). Studying regional and localized watershed processes can allow scientists to analyze and better understand the current state of an area's water resources.

Streams are a critical component for the proper function of Earth's natural systems. Healthy streams promote the health of connecting waterbodies such as rivers, lakes, estuaries, and even oceans. By carrying water, essential nutrients, sediments and other materials, streams help to maintain the ecological health of aquatic and terrestrial ecosystems (Smithsonian Environmental Research Center). "Channel stability" is a term that is frequently referred to in the literature when discussing the ecological function of a stream or river. However, this term can be interpreted in many ways. For the purpose of this study, the Rosen (1996) definition of "channel stability" will be referenced: "is the ability of a stream, over time, in the present climate, to transport the sediment and flows produced by its watershed in such a manner that the stream maintains its dimension, pattern and profile without neither aggrading nor degrading" (Rosgen 2001). Therefore, the stability of a stream channel is critical for maintaining its natural morphology so that it can protect the quality of the water that flows through it, as well as support the life inhabiting its ecosystem. Ultimately, streams support human life and proper ecological function of ecosystems through various services.

Ecosystem services are the processes and conditions through which natural ecosystems support and maintain human life (Daily, 1997). According to the Millennium Ecosystem Assessment, ecosystem services can be categorized into provisioning, regulating, supporting and cultural services. Streams and rivers supply numerous ecosystem services in each of the aforementioned categories such as providing water resources (provisioning), flood control (regulating), habitat (supporting), and recreational use (cultural).

Fresh water and aquatic organisms are resources that are directly provided by streams, which can be consumed or utilized to support human life. Humans rely on streams for the consumptive use of water for domestic, industrial, and agriculture use. Non-consumptive uses of water resources include hydropower generation and transportation. Aquatic organisms such as fish are consumed for food and medicinal purposes. Proper stream function helps to protect these resources by maintaining water quality through natural filtration as well as controlling high flows and erosion.

Streams provide habitat for many freshwater fish species, as well as diadromous fish such as salmon. A diadromous species is one that spends parts of its life in both fresh water and salt water habitats (US Fish and Wildlife Service). Additionally, plants, amphibians and aquatic insects also rely on streams for habitat. Aquatic insects are a crucial part of an ecosystem’s food web and act as indicator species, which help to determine the health of a stream (Smithsonian Environmental Research Institute). For example, some aquatic insects like mayflies are sensitive to pollution-degraded stream habitats, and therefore, their presence often indicates an unpolluted environment. Human population growth and urban development threaten the health of aquatic life, which in turn, adversely effects the overall health of a stream.

The global population has grown by 4 billion people since 1950, and it is projected to rise to 9.2 billion by 2050 (Bongaarts 2009). The rapid population growth of the twentieth century led to a 12-fold increase in carbon dioxide emissions, the leading greenhouse gas contributing to global warming. A primary concern of global climate change is the impact on water supplies (Vorosmarty et al. 2013). Over the last several decades, the amount of precipitation in the United

States has been indicative of a changing climate. In the Southwest, water shortages are becoming more prevalent, but the demand for water continues to rise due to population growth (EPA). This area of the country also experiences more intense and prolonged droughts. The Colorado River water supply is a notable example of a major water source that is threatened by the effects of climate change (EPA). Despite the drought conditions occurring in some parts of the country, the annual average precipitation over the continental United States has increased between 1895 and 2011 (Georgakakos et al. 2014). Additionally, the frequency and intensity of precipitation events has increased across most of the country (Georgakakos et al. 2014). Not only is the amount and type of precipitation largely influenced by a changing climate, but all parts of the water cycle are impacted. For example, evaporation increases under warmer conditions, which increases the amount of water held within the atmosphere. This phenomenon can lead to greater precipitation in some areas, while other areas may experience drier than normal conditions. Climate-related changes in the water cycle have also led to increasing sea level rise, affecting inland waters of coastal areas of the United States. Sea level rise threatens inland freshwater resources through saltwater intrusion. Changes in the salinity of a freshwater ecosystem can drastically alter its habitat, making it unfit for sensitive species. Ultimately, climate change affects the quantity and quality of freshwater resources all over the world. The increasing human population in urban centers is a driving factor of climate change and the variability in water cycle patterns that have been observed in recent decades. Not only does a warming climate alter these patterns, but land use and land cover changes directly affect the water cycle by changing the natural flow, capture and storage of water.

Urbanization and urban sprawl are often the result of population growth in the United States. Urbanization alters the natural flow of stream channels through the construction of impervious surfaces and artificial drainage structures (Meyer et al. 2003). Land use and land cover change are a considerable threat to the ecological health and natural function of streams.

Hydrological processes within a watershed are strongly influenced by the land use and land cover of the area (Schoonover et al. 2006). Impermeable surfaces cover previously vegetated areas and are usually artificial structures that water cannot penetrate. Therefore, impervious surfaces like roads, roofs, or parking lots decrease the infiltration of water into the ground and increase overland flow.

Overland flow, also known as runoff, increases with storm duration and intensity as well as the concentration of impervious surfaces in an area. To account for the loss of natural, permeable land, artificial drainage structures like storm sewers must be installed in urban areas to redirect overland flow. Storm sewers discharge runoff into nearby streams, causing them to receive runoff at a much faster rate and greater volume than watersheds with more natural cover (Meyer et al. 2003). The excess water discharged by the storm sewer increases the speed of streamflow, which ultimately affects the stability of the downstream channels. Faster moving streamflow decreases the ability of the streambed and banks to absorb the water. Additionally, rapid streamflow increases erosion and degrades the natural shape of a stream channel (Meyer et al. 2003). Ultimately, an increase of impervious surface near headwater streams can drastically affect the stability of downstream areas, making them more susceptible to large, frequent floods.

Urbanization is not a single action, but a collection of actions that alter a landscape from its natural condition. Anthropogenic changes are not only made to the land area of an urban watershed, but directly to its streams as well. Stream alterations affect the hydrologic function and physical structure of a channel, and they are often driven by economic advancement. For example, resources such as fish or hydro-electric power are extracted, or goods are transported over these waterways (Yeakley et al. 2016). Most often, changes driven by economic factors shift the balance of ecosystem services and place a greater value on the material outputs of a stream. This shift in balance lessens the value of the regulating and supporting ecosystem services, leaving a stream more susceptible to threats like flooding, erosion and habitat loss. In addition to urbanization, agricultural land use affects the health of aquatic ecosystems.

Agricultural land use is a primary source of sediment. Excess sediment can severely degrade the habitats of fish and macroinvertebrates, causing disruptions in the food web and decreasing ecosystem biodiversity. As previously mentioned, activities that occur upstream have an effect on the downstream water quality. Therefore, erosion that occurs in upstream areas can cause a build-up of sediment deposits downstream, increasing habitat degradation and changing stream morphology in those areas as well. Aside from excess sediment, urban and agricultural runoff can also degrade stream habitats through various nutrients and other contaminants.

A leading cause of impairment to freshwater ecosystems is eutrophication (Chislock et al., 2013; WRI, 2018). Eutrophication is a process that naturally occurs over long periods of geologic time, and is caused by an intake of excess nutrients and sediment into a waterbody. (ESA, 1998). However, human activities have accelerated the extent and rate of this process, causing harmful freshwater pollution. It was not until the mid-20th century that eutrophication was observed as a type of pollution in North American lakes and reservoirs (Science Daily 2018). Eutrophication causes algal blooms and oxygen-depleted (hypoxic) conditions (WRI 2018). Nitrogen and phosphorus from urban and agricultural runoff are primary contributors to the nutrient-rich conditions associated with eutrophication. This can lead to harmful consequences for fisheries (such as fish kills), recreational water bodies, and drinking water sources (Chislock et al. 2013).

Several management techniques have been suggested for the improvement of water quality in streams and reservoirs, especially those susceptible to eutrophication. These techniques include, but are not limited to, artificial mixing and oxygenation, sediment removal, algicides, biomanipulation (fish management), and light reduction (Straškraba & Tundisi, 1999). However, many of these management strategies are implemented as a measure to treat the symptoms of eutrophication, but not the cause. A widely used management technique for the prevention of water pollution is the regulation of riparian buffer zones.

Riparian buffers are strips of forested or vegetated land adjacent to streams and rivers, and are proven effective in mitigating nonpoint source pollution by intercepting and reducing nutrient and sediment loads before it enters a water body (Lowrance et al. 2000, Mayer et al. 2007). In addition to being a natural filtration system, riparian buffers are useful in controlling erosion, providing flood control, moderating water temperature, and providing habitat for wildlife (NCDEQ).

According to the North Carolina Forest Service, Best Management Practices (BMPs) are "a practice, or a combination of practices that is determined to be an effective and practicable means of preventing or reducing the amount of pollution generated by nonpoint sources to a level compatible with water quality goals" (NCFS 2017). Riparian buffers are critical for protecting the water quality of rivers, streams and lakes, and are consistently recommended in the literature as a BMP for preventing watershed pollution (Lowarance et al. 2000). The implementation of riparian buffers is one example of an effective stream restoration practice that focuses on the stability and health of North Carolina surface waters. Planting trees in the buffer zone is an essential part of stream restoration, and their growth is monitored for years following construction. A measure of the overall success of a restoration project is the number of trees that survive and the extent of their growth.

In 1972, The Clean Water Act was passed as a federal law in the United States in order to regulate surface water pollution and water quality standards (EPA 2018). In section 303(d) of the Clean Water Act, it states that impaired and threatened waters are those that do not meet the water quality standards, and must be identified by individual states. By submitting a list of impaired and threatened surface waters to the Environmental Protection Agency (EPA) every two years, states must also commit to developing Total Maximum Daily Loads (TMDL) in order to address the pollutant(s) causing impairment (EPA 2018). A TMDL is the maximum amount of a pollutant that can exist in an impaired waterbody, while still meeting water quality standards. The most important first step in stream restoration is eliminating the source of impairment. Regulations like the Clean Water Act aim to identify and address sources of water quality impairment, ultimately restoring surface waters to a more natural state. However, mitigating sources of pollution does not always restore a stream to its natural state. As previously discussed, streams that receive a lot of runoff from urban and agricultural areas have been impaired by erosion and sedimentation. Therefore, additional restoration efforts that focus on the physical structure and hydrologic processes of degraded streams are needed.

Federal, state, local and private organizations are proposing restoration projects in order to improve the stability and ecological health of impaired streams (White 2001). Modern restoration practices implement natural channel design methods, which place an emphasis on working with the natural processes of a stream, instead of fighting them, to repair the damage (White 2001). Stream channel stability depends on whether or not a stream is able to maintain its natural dimensions, flow pattern and slope (Doll et al.). Therefore, natural channel stability can be observed in streams that can naturally transport the sediment that is deposited from runoff or upstream areas of the watershed. For proper hydrologic function and ecological health, it is important for streams to maintain their form over time. In restoration, the instability of a stream is addressed through natural channel design methods. The model, or template that is used as a guide for restoring a degraded stream is called a reference reach.

In order to restore the balance between economic growth from land use development and ecological health, environmental mitigation practices are widely used across the United States. Government agencies and the environmental industry define mitigation as a way to offset the known impacts on streams, wetlands and other natural resources. Mitigation is usually regulated through a crediting system in which different values are placed on different types of mitigation. In the United States, projects are classified as preservation, enhancement, restoration or creation (PERC) of a natural resource. For example, more credits are rewarded for restoration of a stream than preservation of a stream. Based on the project, multiple types of mitigation are allowable. This allows mitigation companies to maximize the number of credits at a project site, while also maximizing ecological benefit. The mitigation industry is somewhat unique, in that governing bodies work closely with private entities to manage the crediting system. The governing bodies form the ecological criteria that defines the value of a mitigation credit, and the environmental companies must comply to receive the credits. For example, the North Carolina Division of Mitigation Services (DMS), which is a division of the North Carolina Department of Environmental Quality (DEQ), makes contractual agreements with private environmental companies in North Carolina to complete mitigation projects that are valued at a certain number of credits. At different levels of completion of a project, the environmental company may be rewarded certain allocations of the total number of credits. Mitigation provides an alternative to the obstacle of certain environmental laws by allowing developers who impact natural resources to purchase these credits. Stream restoration is one of the most common types of environmental mitigation projects.

As previously mentioned, state governments primarily regulate the mitigation process, so it may differ across states. In North Carolina, the process begins as the DMS releases a request for proposal (RFP) stating that they are in need of a certain amount of mitigation credits. Then, mitigation providers look for potential project sites that will fulfill that need. Often, mitigation companies have a ″land″ team who is responsible for finding these sites, contacting landowners, and setting up easements. Once the providers submit their proposals, the DMS will choose the best site for the lowest price and make a contractual agreement with that provider. Next, the provider writes a mitigation plan and submits it to the Interagency Review Team (IRT). The NC IRT consists of representatives from the U.S. Army Corps of Engineers (USACE), the U.S. Fish and Wildlife Service (USFWS), NC Department of Transportation (NCDOT), DMS, and others. The IRT has a maximum of ninety days to review the mitigation pan and make comments. Once the comments are returned to the provider, they are not bound by a certain timeframe in which the revisions must be made and sent to the IRT. However, it is in the best interest of the provider to address all of the comments as quickly as possible in order to keep the project moving forward. Once the final mitigation plan is submitted, the IRT then has another thirty days to approve/disapprove it. When the mitigation plan is approved, the provider will begin construction of the project. In order to properly restore a stream, ecological restoration designers must be knowledgeable of its fluvial geomorphology, which includes its natural function and interactions with the surrounding landscape. The formation, function and landscape interactions of streams are constantly changing, but an understanding of these morphological characteristics can aid in determining stream stability and response to stressors. Stressors that can reduce the stability of a stream system include climate change, land use/ land cover change, and hydrologic structures such as dams. These stressors all affect the streamflow and/or a stream′s sediment load in some way, which ultimately changes its morphology. Loss of stability in a stream system not only alters its physical characteristics, but also affects the overall ecological health of a stream. For example, when morphological changes occur, a loss of biological diversity within the stream ecosystem can also occur due to habitat alteration or habitat loss. In fluvial geomorphology, natural channel design techniques are often used for stream restoration in order to create stable channels and increase stream functions as much as possible at any given site (Harman and Starr, 2011). Establishing a design criterion for a restoration project is one of the most critical components of a successful natural channel design. The most common method of establishing numerical criteria for channel design characteristics such as dimension, pattern and profile is the reference reach survey approach (Rosgen 1998; Harman and Starr 2011).

As first described, a reference reach is a stable channel segment that serves as a model for restoration, and is described by quantitative field data of its morphological features (Rosgen 1998). The Rosgen Classification of Natural Rivers (1994) allows stream reaches to be grouped into categories according to their morphological features, which include entrenchment ratio, sinuosity, width/depth ratio, slope and channel materials (Rosgen 1994). A reference reach can be located on the same channel, but on a different reach than the restoration project, or on a different channel that has similar watershed characteristics as the restoration project (White 2001). A reference reach does not have to be a pristine stream, but it must display a stable morphology within its environment (Doll et al.). When selecting an appropriate reference reach, several factors are considered.

Generally, environmental mitigation providers only have a small selection of reference reach sites to choose from for each project. Finding new reference reach sites can be time consuming because a field survey is required to obtain the morphological parameter values. Before the field survey is performed, they must search for a potential site (i.e. with Google Earth), and drive to its location. Therefore, due to the fast-paced and demanding nature of the mitigation industry, surveying potential new reference reach sites is often a low priority.

Reference reaches are often located in areas with a low density of urban land use. A reference reach should be in a stable watershed that has not experienced major land use changes in recent years (Doll et al.). Reference reaches are also characterized by having a bankfull stage that reaches the top of the bank, stream banks with gradual sloping, well-vegetated banks, and no erosion of the banks or bed (Doll et al.). At least one reference reach should be used for each restoration design, but a better overall assessment of morphologic relationships can be made by using several reference reaches.

For complex restoration projects, it is best if multiple methods are used to develop a final set of design criteria. Ultimately, professional judgement is required to select the final criteria, which is why design experience is important. Many designers, for example, rely solely on reference reaches to develop their design criteria. The reference reach approach requires that the appropriate stream type be designed for the given valley type, geology and land use. If the valley is confined, for instance, the approach dictates that a ‘Bc’ stream type should be designed. Also, the pre-existing stream type may be different than the proposed stream type, i.e., the existing stream was a ‘F4’, but the proposed channel is a ‘B4c’ because of channel confinement caused by lateral constraints. While this is an acceptable approach, there are limitations. First, reference reaches are difficult to find in many parts of the United States that have experienced urban and suburban growth. Second, most reference reaches in the east are found in mature bottomland hardwood forests where the pattern has been primarily dictated by large trees. In other words, these streams are not free to form their pattern. Resulting pattern ratios are not suitable for design projects, which are often constructed in valleys stripped of woody vegetation. This is why reference reach ratios should be compared to evaluation results from past projects and why multiple techniques for developing design criteria should be used.

Ultimately, many of the design ratios will be different from the reference reach ratios due to site conditions. For example, the radius of curvature ratio, bankfull width/depth ratio, pool width ratio, meander width ratio, etc. are adjusted to create a design that can evolve towards the reference condition over time. This is necessary because project sites often lack floodplain vegetation, whereas the reference reach is within a mature forest. These adjustments allow the stream to evolve towards the reference condition over time as the buffer becomes established.

In a 2001 U.S. Geological Survey publication (White 2001), several criteria were laid out in order to find an appropriate reference reach in a non-urban, piedmont region of Pennsylvania and Maryland. The reference reach criteria used in this study include: runoff characteristics that are similar to those of the restoration project site; channel width and depth within the reach do not change drastically; rapidly eroding banks or areas of excessive deposition are not evident; riffles are characterized by an appropriate degree of embeddedness; and accumulation of debris is not excessive (White 2001). The U.S. Geological Survey study conducts a thorough reference reach search with all of the necessary morphological elements needed for an efficient natural channel design approach (White 2001). Ultimately, finding an appropriate reference reach for a stream restoration project can be time-consuming, especially when looking in a watershed where much of the natural landscape has been converted to agricultural or urban land uses.

Analyzing the suitability of a new reference reach site requires an extensive field survey in order to determine certain morphological parameters that cannot be determined from the office. Therefore, finding new reference reach sites for every stream restoration project is inefficient and impractical. To expedite the design process, stream restoration designers often reuse the same reference reaches for several projects. Although many reference reaches can serve as valid stream design models for multiple projects within the same geographic region, they might not be the best fit when considering certain stream reach characteristics. The purpose of this project is to make the identification of reference reaches easier by creating a Geographic Information Systems (GIS)- based tool. We believe that the Reference Reach Finder Tool will improve the stream restoration design process and have important benefits to natural ecosystems..

2.0 Methods

2.1 Database Establishment

A database of reference reach sites in North Carolina was established to be used as the central basis for the Reference Reach Finder Tool. This database was used for all testing of the Python code and general execution of the tool's features. An extensive procedure was formulated to ensure that the included data is viable, relevant, and accurate.

2.1.1 Reference Reach Site Locations

Before finding reference reach locations, a timeframe was established in order to narrow the search. Only North Carolina Department of Mitigation Services (DMS) mitigation plans released from 2009 to May 2018 were used in this project to reduce the likelihood that major morphological changes of their reference reach sites have occurred. Establishing the timeframe for data collection narrowed the search from almost 700 mitigation plans to under 300 mitigation plans. Each qualifying mitigation plan was searched to find the name(s) of the reference reach(es) used to aid in the design of the project, as well as a description of its (their) geographic location(s). Some mitigation projects used several reference reaches to aid in design, and others used only one. Often, reference reaches were used in several different mitigation plans as well.

In Google Earth, a point was placed in the best-estimated location for each reference reach. Detailed mitigation plans included a map with the geographic location(s) of the reference reach(es) as well as a written description of the location. Less detailed mitigation plans did not include a map or a detailed written description of the location, but rather a vague idea of where the reference reach(es) is (are) located. For example, a mitigation plan might have described its reference reach as being five miles southwest of a certain road, or in the northeast region of Orange County, NC. These vague descriptions made it challenging to find reference reaches in Google Earth. The EPA's WATERS layer was added to Google Earth and used to guide the search for existing reference reaches. Specifically, the streams surface water feature from this dataset was turned on during this process. The streams feature layer includes stream names, which were helpful for identifying many of the reference reaches.

Employees from the North Carolina Department of Environmental Quality (NCDEQ) were contacted in order to explore the potential for expanding the aforementioned reference reach database. A geodatabase with a reference reach feature class (line type), an associated mitigation project feature class, and a relationship class were obtained through this effort. The relationship class allowed for matching between the reference reaches and the mitigation projects that used them for design guidance. Many of the additions to the reference reach database from NC DEQ are older than the initial timeframe set for the search criteria of this project. Additionally, many of the associated mitigation plans were unavailable to be viewed, some of the relationship class data was mismatched, and some of the reference reaches were already recorded in the original database. The usable data from the NC DEQ was included in the reference reach database table (**Appendix B**).

Based on the estimated pinned locations found in Google Earth, a master list of eighty reference reach sites was created, which can be found in **Appendix A**. The master site list lists the mitigation plan(s) where each reference reach can be found. Next, an organizational file directory was created that includes a folder for each reference reach containing the corresponding mitigation plan(s) document(s). The reference reach master site list and directory were created as organizational tools to help with the next step of database development.

2.1.2 Morphological Data Collection

The mitigation plans were searched through a second time in order to find the morphological data tables for each reference reach. When available, data was collected for the Rosgen stream type, drainage area, valley slope and D50/substrate for each reference reach and then compiled into a spreadsheet. This spreadsheet is the foundation of the reference reach database. The page numbers of the morphological tables within each mitigation plan are noted on the master site list for each reference reach in order to easily access them in the future (**Appendix A**). The process of searching through the mitigation plans for morphological data was repeated for the NC DEQ reference reach data once it was obtained.

2.1.3 GIS Database Development

The pinned locations of the reference reach sites within Google Earth were exported as a KMZ extension file. Within ArcMap, the "KMZ to Layer" tool was executed in order to convert the Google Earth points file to a Shapefile (.shp). Editing, data analysis, and other capabilities are available when using a Shapefile within ArcMap.

The reference reach database spreadsheet was used to create the attribute table for the reference reach point Shapefile in ArcMap. The relevant reference reach feature class data obtained from the NC DEQ was combined with the original Shapefile. The DEQ feature class is line type data, and therefore the chosen reaches were edited into the original Shapefile as points in order to maintain consistency.

2.2 Interface Design

The Reference Reach Finder Tool is implemented in the form of a graphical user interface (GUI). A GUI allows the user to easily interact with the tool by eliminating direct user interaction with the Python script. Instead, the tool will be displayed to the user in a categorized series of windows that provide instructions for user input (Tateosian 2015). The GUI is created by including code for its design within the Python Toolbox script. The Python Toolbox will be explained in greater detail in section **2.4.4**.

2.2.1 Original Tool Design

The original design interface for the Reference Reach Finder Tool was hand-drawn, and is shown in **Appendix C**. The original design was drawn before the final Python script was written or tested, therefore, some of the features included in the original design were unable to be implemented in the tool's final GUI. The limitations of the GUI are discussed in section **4.3** of the **Discussion and Conclusion**.

2.2.2 Final Tool Design

The final tool design is the Python-coded version of the original tool design. A Python Toolbox was used to create the GUI due to the large number of parameters involved. In a Python

Toolbox, the parameter properties are written directly in the tool's Python code, whereas in a Script Tool, the parameter properties must be set up in the Script Tool Wizard. Storing all of the parameter information in the same Python file as the rest of the tool's code streamlines the tool-making process, and eliminates the need to navigate back and forth between the Python code and the Script Tool Wizard. Additionally, the Python Toolbox method makes it easier to pinpoint errors in the code and debug them(see section **2.3.4**). A categorized design was implemented in the code in order to organize the parameters within the GUI, making it easier to navigate and more visually appealing. There are seven categories, and each category is a different step that the user can choose to take. The steps are numbered by the suggested order that they should be followed when using the tool. The final GUI design for the Reference Reach Finder Tool is shown in the **Results** section of the report. The following snippet of code within the Python Toolbox script is responsible for creating the categorized design:

```
    for index in range(numParams):
        if index in range(3, 7):
            paramList[index].category = '1. River Basin Selection'
        elif index in range(7, 14):
           paramList[index].category = '2. Stream Type Selection/ Unknown Stream Type Determination'

        elif index in range(14, 18):
            paramList[index].category = '3. Drainage Area Entry'
        elif index in range(18, 29):
            paramList[index].category = '4. Drainage Area Calculator'
        elif index in range(29, 33):
                   paramList[index].category = '5. Valley Slope Entry'
        elif index in range(33, 41):
            paramList[index].category = '6. Valley Slope Calculator'
        elif index in range(41, 44):
            paramList[index].category = '7. Median Grain Size (D50) Selection'
```

2.3 Coding

2.3.1 Python for ArcGIS

The programming language that is used to write the script for the Reference Reach Finder Tool is called Python. Python is an object-oriented programming (OOP) language, and everything in Python is an object with properties and associated functions (Tateosian 2015). Python is often used in the geospatial field to automate data analysis processes. Python code can be run directly within ArcGIS software, allowing for many geospatial data analysis procedures to executed. Python is useful for performing batch processing on a group of files, which saves time by eliminating the need to repeat a certain task for each file within the group. The Python script created for the Reference Reach Finder Tool is required to be run within ArcGIS due to software-specific capabilities included in the code.

2.3.2 Pseudocode and Workflow Charts

Pseudocode is used by programmers to control the workflow of a Python script. Pseudocode is particularly helpful when writing complex scripts that have several nested loops. It allows the programmer to plan the script in terms that are similar to writing in "plain English" rather than Python code. Once the flow of the script is outlined, it is then easier for the programmer to focus on the syntax details (Tateosian 2015). A pseudocode for the Reference Reach Finder Tool was written before the Python code for this purpose.

When writing a complex Python script, it is also useful to map out the desired workflow using visual aids such as charts. Workflow charts are particularly useful when communicating the inputs, outputs, and overall goals of a script to an audience. A basic overview of the workflow for the Reference Reach Finder tool is shown in **Figure 1**. Additionally, **Figure 2** shows the drainage area calculator feature process that is implemented in the script. These workflow charts show the data input at the top, the execution of the processes and tools in the middle, and the final output at the bottom.

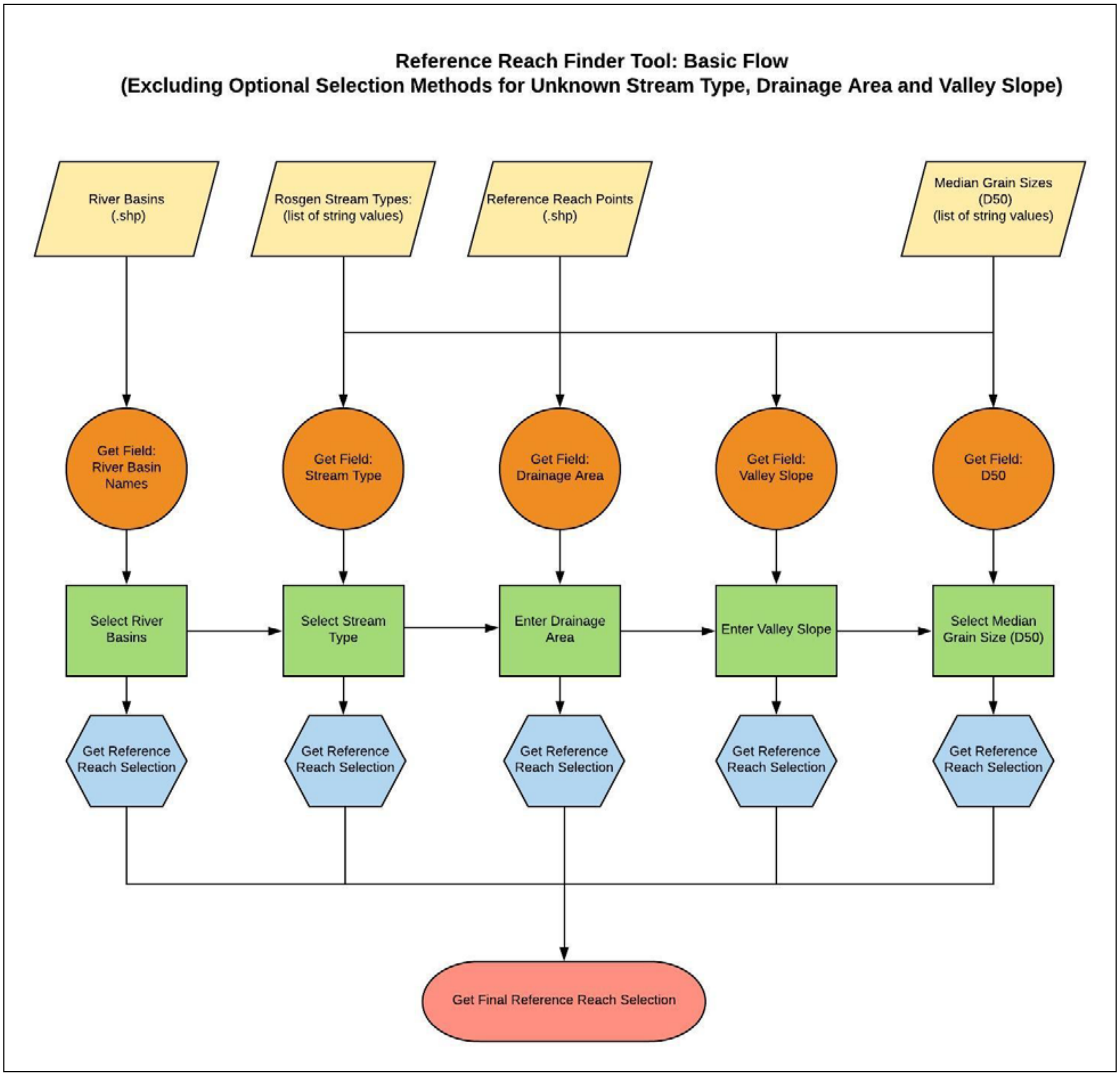

Figure 1. *Flow diagram of important steps in the Reference Reach Finder Tool workflow.*

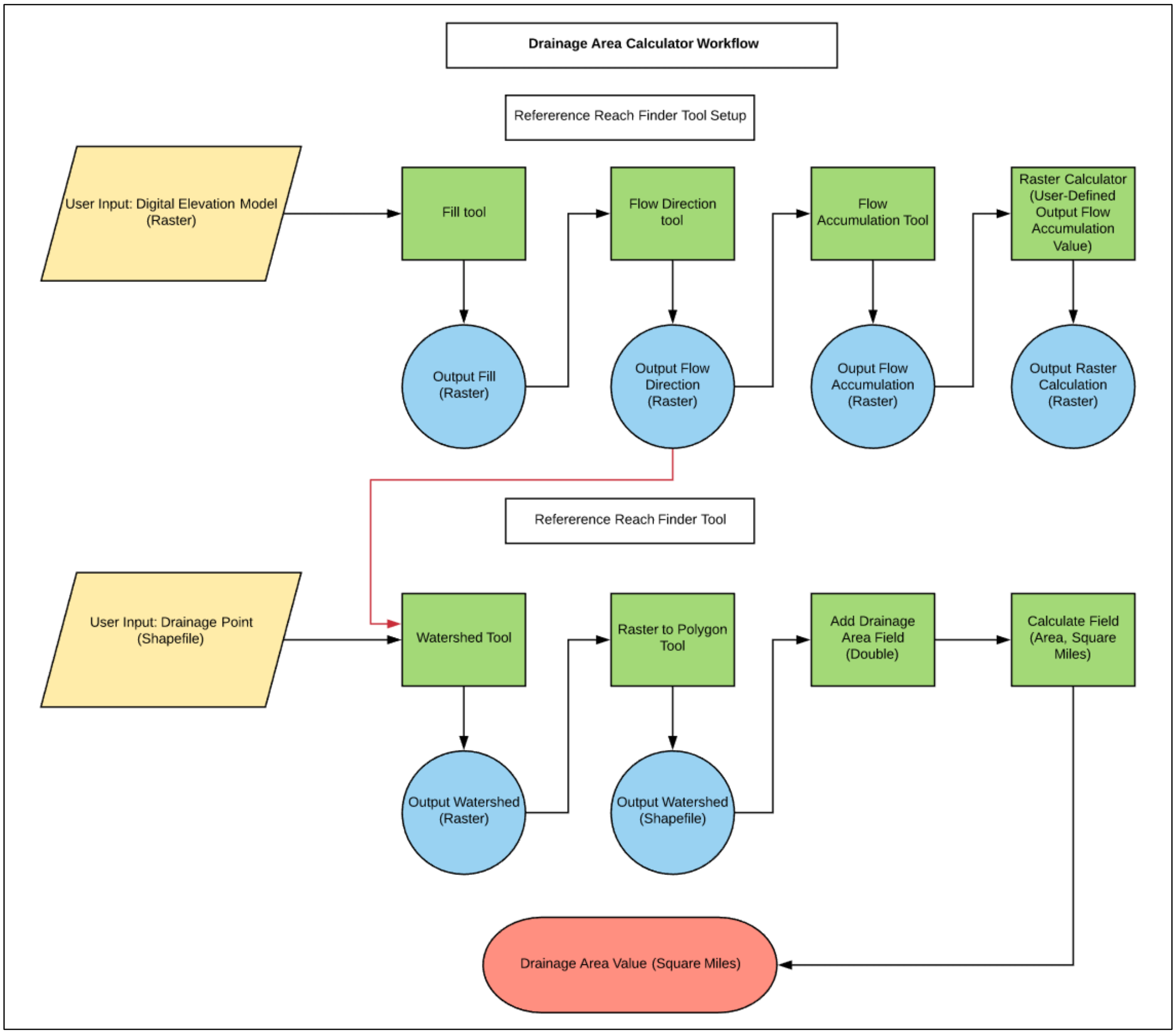

Figure 2. *Flow diagram of the important steps in the drainage area calculator feature within the Reference Reach Finder Tool code.*

2.3.3 Python Code

A Python script was written in order to execute the Reference Reach Finder Tool within ArcGIS software. An integrated development environment (IDE) is a software application that is used for computer programming (Tateosian 2015). PythonWin is the IDE that was used to develop the script for the Reference Reach Finder Tool. The PythonWin software provides a user-friendly way to check Python syntax, color-code different types of statements, debug problems in the code, and test the code (Tateosian 2015). The PythonWin IDE also has an interactive window that allows the user to check individual lines of code without running the entire script. To learn how to code in Python, a semester-long course was taken. Geospatial Programming (GIS 540) is currently a course at North Carolina State University, taught by Dr. Laura Tateosian in the Center for Geospatial Analytics, that teaches students how to code in Python for the primary purpose of automating processes in ArcGIS software. The complete Reference Reach Finder Tool Python script is shown in **Appendix D**.

2.3.4 Python Toolbox

A Python Toolbox was used to set up the GUI for the Reference Reach Finder Tool. A Python Toolbox (.pyt) is an ASCII based file that is created entirely in Python (Tateosian 2015, ArcGIS Resources). The Python code of a Python Toolbox defines a toolbox and its tools (Tateosian 2015, ArcGIS Resources). In terms of set up, a Script Tool is different than a Python Toolbox because the Script Tool wizard is used to set up the Script Tool properties. Therefore, when creating a Script Tool, the parameter properties are specified separately from the associated Python Code in the 'Parameters' tab of the Script Tool Wizard (Tateosian 2015). The Script Tool wizard allows the user to explore the names of parameter types, properties and types of GUIs that can be created, therefore, it is a good learning tool for programmers who would like to transition to using Python Toolboxes (Tateosian 2015). As previously mentioned, Python Toolboxes are completely Python based, and therefore, the parameter properties are set up directly in the associated Python script, rather than in a Script Tool wizard. Within a Python Toolbox script, there is a toolbox class and a tool class. The toolbox class defines the toolbox, and the tool class contains six different functions. The tool class functions include initializing the tool name, defining parameters, identify proper licensing for tool execution, updating parameters before internal validation, updating messages and executing the source code of the tool (Tateosian 2015). Each parameter and its properties were written inside the ‘getParameterInfo’ function. For example, the following code defines the parameter properties for the input reference reach points Shapefile:

```
        # Select reference reach point shapefile
        param2 = arcpy.Parameter()
        param2.name = 'Reference_Reach_Points_File'
        param2.displayName = 'Select Reference Reach Points File'
        param2.parameterType = 'Required'
        param2.direction = 'Input'
        param2.datatype = 'DEShapefile'
        relativePath1 = '../data/NC_Reference_Reaches/NC_Reference_Reach_Points.shp'
        param2.value = getAbsPath(relativePath1)
        param2.filter.list = ['.shp']
```

Forty-five parameters were defined inside of this function for the main tool.

The ‘isLicensed’ function was used to identify whether or not the Spatial Analyst extension is available to the user because it required for several tools to execute properly when the Reference Reach Finder Tool is run. The following code achieves the Spatial Analyst license check:

```
        if arcpy.CheckExtension('Spatial') == 'Available':
            return True
        else:
            return False
```

The ‘updateParameters’ function was used to dynamically update the GUI when the stream type determination feature is being used (Section 2.2.2). The following lines of code allow the second and third drop-down menus to populate correctly based on the selection of the first drop-down menu (entrenchment ratio):

```
    if parameters[4].value:
        with arcpy.da.SearchCursor(parameters[3].valueAsText, parameters[4].valueAsText) as rows:

            parameters[5].filter.list = sorted(list(set([row[0] for row in rows])))
    else:
        parameters[5].filter.list = []
```

```
        
    if parameters[10].altered:
        if parameters[10].value == "Slightly Entrenched (Ratio> 2.2)":
            parameters[11].filter.list = ["Very Low Width to Depth Ratio (< 12)", "Moderate to High Width to Depth Ratio (> 12)"]
            parameters[12].filter.list = ["Moderate to High Sinuosity (>  1.2)", "High Sinuosity (> 1.5)"]
            if parameters[11].altered:
                if parameters[11].value == "Very Low Width to Depth Ratio (< 12)":
                    parameters[12].filter.list = ["High Sinuosity (> 1.5)"]
                    if parameters[12].altered:
                        parameters[8].filter.list = ['E']
                        arcpy.AddMessage("Stream Type is E")


                elif parameters[11].value == "Moderate to High Width to Depth Ratio (> 12)":
                    parameters[12].filter.list = ["Moderate to High Sinuosity (>  1.2)"]
                    if parameters[12].altered:
                        parameters[8].filter.list = ['C']
                        arcpy.AddMessage("Stream Type is C")

        elif parameters[10].value == "Moderately Entrenched (Ratio 1.4 - 2.2)":
            parameters[11].filter.list = ["Moderate Width to Depth Ratio (> 12)"]
            parameters[12].filter.list = ["Moderate Sinuosity (>  1.2)"]
            if parameters[11].altered and parameters[2].altered:
                    parameters[8].filter.list = ['B']
                    arcpy.AddMessage("Stream Type is B")

        elif parameters[10].value == "Entrenched (Ratio > 1.4)":
            parameters[11].filter.list = ["Low Width to Depth Ratio (< 12)", "Moderate to High Width to Depth Ratio (> 12)"]
            parameters[12].filter.list = ["Low Sinuosity (< 1.2)", "Moderate Sinuosity (>  1.2)"]

            if parameters[11].altered:
                if parameters[11].value == "Moderate to High Width to Depth Ratio (> 12)":
                    parameters[12].filter.list = ["Moderate Sinuosity (>  1.2)"]
                    if parameters[12].altered:
                        parameters[8].filter.list = ['F']
                        arcpy.AddMessage("Stream Type is F")

                elif parameters[11].value == "Low Width to Depth Ratio (< 12)":
                    if parameters[12].altered:
                        if parameters[12].value == "Low Sinuosity (< 1.2)":
                            arcpy.addMessage("Stream Type is A")
                        if parameters[12].value == "Moderate Sinuosity (>  1.2)":
                            parameters[8].filter.list = ['G']
                            arcpy.AddMessage("Stream Type is G")



    return
```

The ‘execute’ function contains the main source code for the Reference Reach Finder Tool (**Appendix D**).

When the GUI was created using the Python Toolbox, PythonWin was set as the editor environment. Therefore, the Python Toolbox was able to open the script in a separate PythonWin window for editing. This editing method makes the debugging process much easier, especially when the Python Toolbox script is complex and several parameters are involved. Overall, a Python Toolbox is a means of streamlining the GUI making process by storing the GUI properties and the tool's source code in a single, organized Python (.pyt) file (Tateosian 2015).

2.3.4.1 Reference Reach Finder Tool Setup

In addition to the main Reference Reach Finder Tool, a Reference Reach Finder Tool Setup (RRFT-S) tool was added to the Python Toolbox. The RRFT-S allows the user to add the reference reach database layer, river basins with name labels, and contour lines with elevation labels to the map for reference before the main tool is executed. These layers will be helpful for the user to spatially reference before and while using the main tool. The user only needs to input a contour lines file if they would like to use the 'Valley Slope Calculator' feature within the main tool. Adding the contour lines to the map allows the user to find the upstream and downstream valley elevation values, and the length of the valley of the stream of interest. Additionally, the Setup tool will expedite the watershed delineation process for the 'Drainage Area Calculator' feature in the main tool by performing most of the DEM processing. The Fill, Flow Direction and Flow Accumulation tools will all be executed if the user opts to include an input DEM file. If this step is taken, a raster calculation will also be performed using the output flow accumulation raster. A box in the GUI is included for the user to enter a flow accumulation value (i.e. 'Value > 500') to create a raster showing the flowlines of interest. The output calculated raster can be used as a guide for the placement of a drainage point (if user does not yet have a drainage point Shapefile), which can be used as the input drainage point in the main RRFT.

2.4 Parameter Filtering Criteria

Each of the five main parameters in the Reference Reach Finder Tool have unique selection methods and rules for filtering through the attribute table of the reference reach database. The flow of the tool is set up categorically, and all categories, or parameters, included within the tool are currently set as optional. The final selection of reference reach sites is a combination of the selection results of all of the parameters that the user has chosen to include. The code is written so that the final output does not contain duplicates of reference reach points. In order to increase tool flexibility, the user has the option to use the North Carolina data that is included with the tool's data package or their own data. Greater tool flexibility increases the potential of a wider userbase, and reduces limitations on tool improvements in the future. The general rules, selection methods, selection options, and outputs implemented in the tool are listed below for each parameter. Parameter-specific justifications for the rules and selection methods implemented in the Reference Reach Finder Tool are discussed in the **Discussion and Conclusion** section of this report.

2.4.1 River Basin (Step 1)

- **Description:** According to the NC Department of Transportation, reference reaches are generally located within the same physiographic and climatic region as the project site (2003). North Carolina's diverse landscape exhibits a wide range of elevations (mountains, foothills, piedmont, sand hills, coast, etc.). There are seventeen major river basins in NC, and they are all topographically and morphologically diverse. Therefore, the river basin selection is set as the first step within the tool to allow the user to narrow down the search to a geographic region that is at least somewhat topographically/ hydrologically similar to the restoration project of interest.

- **Rule:** Reference reaches only within the selected river basin(s) should be used in output.
- **Use:** Using a drop-down menu, user will select the river basin(s) in which the project(s) of interest is located. The number of river basins varies by state. The North Carolina river basins are used as the default input for this parameter. For example, the seventeen NC river basins listed below will appear as default values within the tool:
    - Cape Fear, Yadkin-Pee Dee, Tar-Pamlico, Neuse, Roanoke, Pasquotank, Lumber, Catawba, French Broad, Little Tennessee, Broad, White Oak, Chowan, New, Hiwassee, Watauga, Savannah.
    - **Option**: User may choose to select multiple river basins.
- **Output:** All reference reaches within selected basin(s) will be selected and returned as output for this parameter.

2.4.2 Rosgen Stream Type (Step 2)

- Description: The Rosgen stream type letter encompasses entrenchment ratio, width/depth ratio and sinuosity characteristics. The Rosgen classification system for natural rivers (**Figure 3**) is a widely used guide for easily distinguishing streams by these morphological dimensions. Each stream type letter has a different combination of these values, and therefore, the stream type is a good method for choosing an appropriate reference reach site. Only the first letters of the Rosgen classification system are available for user selection in the RRFT for simplicity. There are several sub-classifications in the Rosgen key which require the user to know the slope and the channel material, both of which will be included as separate parameter values within the Reference Reach Finder tool.

- **Rule:** Reference reaches must be an exact match to the primary classification level of Rosgen stream types (first letter) for single-thread channels.
- **Use:** Using a drop-down menu, the user will select the 1st level Rosgen stream type for single-thread channels from the following list: A, G, F, B, E, C
- **Option:** If the best stream type of a single-thread channel is unknown, the user may enter information into the tool about the entrenchment ratio, width/depth ratio, and the sinuosity in order to determine the most appropriate stream type. Based on the selection of the first stream type determination parameter (entrenchment ratio), the tool will dynamically update the drop-down lists of the subsequent parameters. This feature guides the user to select a valid combination of stream type determination parameters, which ultimately narrows down the selection to one Rosgen Stream Type letter. The user then knows which letter to select from the original list. For example, if the user selects "Moderately entrenched" from the first category, "Moderate width to depth" and "Moderate sinuosity" will be the only selectable options in the remaining two categories, respectively. This will update the original list of stream types to display only type "B". This tool feature is based on *The Key to the Rosgen Classification of Natural Rivers*, shown in **Figure 1**. The three stream type determination categories and their values are listed below:
    - Entrenchment ratio selection options (drop-down menu):
        - Entrenched (Ratio < 1.4)
        - Moderately entrenched (Ratio 1.4 – 2.2)

        - Slightly entrenched (Ratio > 2.2)
    - Width/depth ratio selection options (drop-down menu):
        - Very low W/D (Ratio < 12)
        - Low W/D (Ratio < 12)
        - Moderate W/D (Ratio > 12)
        - Moderate to high W/D (Ratio > 12)
    - Sinuosity selection options (drop-down menu):
        - Low sinuosity (< 1.2)
        - Moderate sinuosity (> 1.2)
        - Moderate to high sinuosity (> 1.2)
        - High sinuosity (> 1.5)
- **Output:** All reference reach sites that match the selected Rosgen stream type classification letter will be used in determining the final output.

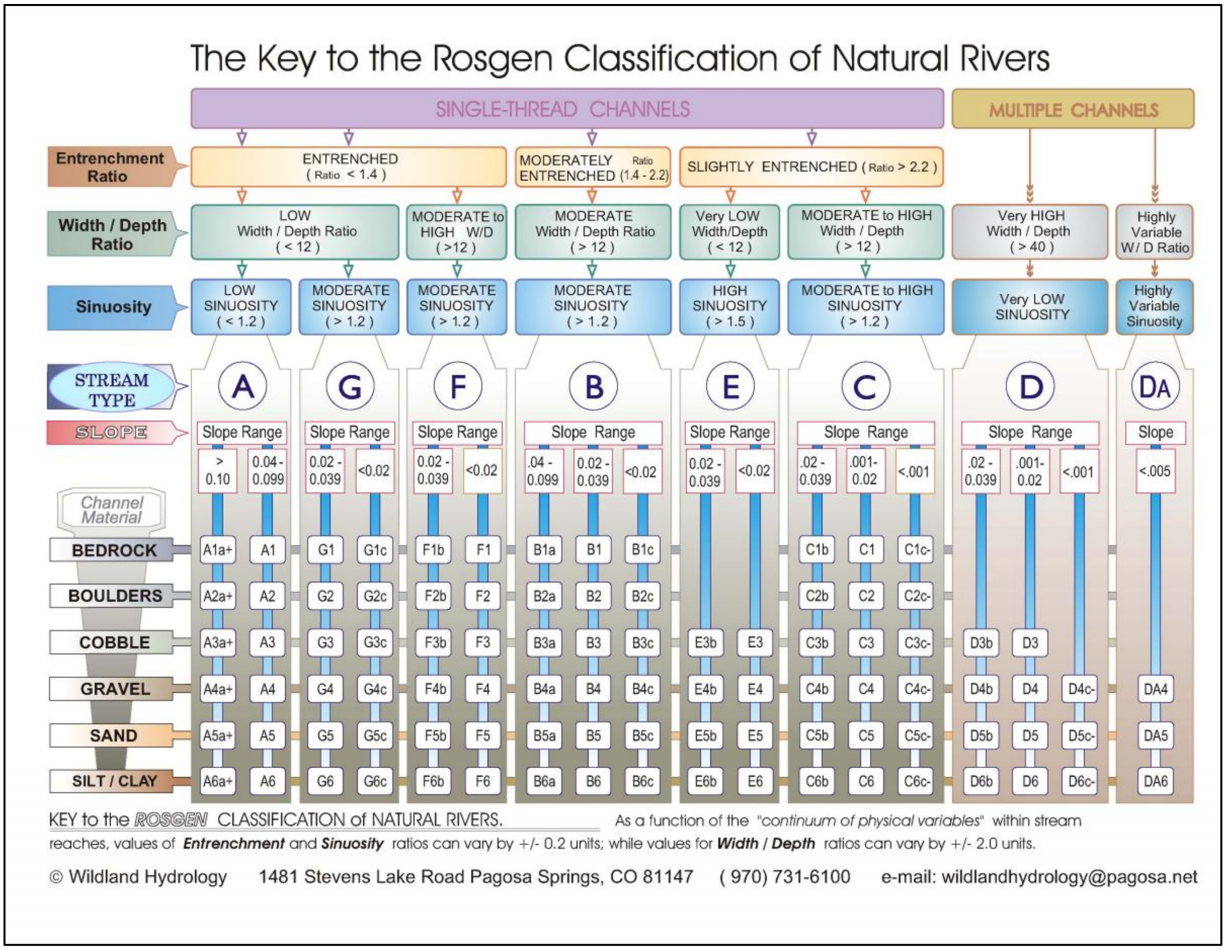

Figure 3*: Key to the Rosgen Classification of Natural Rivers. The Reference Reach Finder tool will determine stream types based on this classification system. Source: Wildland Hydrology.*

2.4.3 Drainage Area (Step 3)

- **Description**: A stream's drainage area is related to its function, structure and health. Any point along a stream network can be considered an outlet (or drainage) point when evaluating the amount of area that drains through that point. All of the land area and streams that drain to that point are considered to be part of its drainage (or watershed) area. The larger the drainage area of a stream, the more water and other materials, like sediment and pollutants, it receives from upstream. Because of this, smaller streams tend to have smaller drainage areas, and larger streams tend to have larger drainage areas. The morphological features, stability and streambed material are more likely to be similar in streams with similar drainage areas.

Therefore, an appropriate reference reach site would have a drainage area value that is similar to that of the restoration project of interest.

- **Rule:** Reference reaches must be within pre-determined ranges of the drainage area of the restoration project of interest. **Table 1** indicates the ranges that will be acceptable for reference reach selection within the database based on the project drainage area input by the user.
- **Use:** User will manually enter a numerical drainage area value (mi^2) for the restoration project of interest.

Table 1: *Reference Reach Finder tool drainage area (mi^2) matching criteria.*

Project Drainage Area (mi^2)	**RR Matching Range (mi^2)**
< 1.0	+/- 0.3
1.0 – <1.5	+/- 0.4
1.5 – <3.0	+/- 0.5
3.0 – <5.0	+/- 1.0
5.0 – <10.0	+/- 2.0
10.0 – <20.0	+/- 4.0
20.0 - <30.0	+/- 5.0

- **Option:** When an appropriate drainage area value is unknown, the tool can calculate it using the "Drainage Area Calculator" feature.
    - **Use:** The user must provide a drainage point file and digital elevation model (DEM) of the area of interest as input. As shown in the workflow chart in **Figure 3**, the tool will run through a series of DEM terrain processing steps using tools in the Hydrology toolset within the Spatial Analyst toolbox in ArcMap. These tools include

Fill, Flow Direction, Flow Accumulation and Watershed. Ultimately, the tool performs a watershed delineation using the drainage point and DEM files.

The ArcMap Raster Calculator tool cannot be used when coding in Python. The Arcpy module that comes built-in with Python software does not have this functionality. When performing a watershed delineation in ArcMap, the Raster Calculator tool is often used with the output Flow Accumulation raster as input. Additionally, an expression is required to run the Raster Calculator. Due to the aforementioned setback in coding the watershed delineation workflow, a workaround was established. The following lines of code perform the raster calculation:

```
rastCalc = arcpy.sa.Con(flowAcc, 0, 1, "Value > 100")
rastCalc.save(workspace + parameters[23].valueAsText)
outRastCalc = arcpy.mapping.Layer(parameters[23].valueAsText)
arcpy.mapping.AddLayer(df, outRastCalc)
```

The expression "Value > 100" produces an output flow accumulation raster that only shows the cells on the input raster that have accumulation values of greater than 100. This method creates a flow accumulation raster with flowpaths that are more defined and easier to see when compared to the original flow accumulation raster. This value was somewhat arbitrarily chosen as the default for the tool after testing different flow accumulation values on a few different rasters using the Raster Calculator tool. A value of 100 produces a raster with the primary flowpaths and many branching smaller-order streams. The greater the value, the less detail in the flowpaths of the output raster, and vice versa. Therefore, a value of 500 would eliminate many of the smaller-order streams, but may make it easier to decide on the placement of a drainage point.

Once the watershed delineation is complete, the Raster to Polygon tool is implemented, with the watershed raster as the input. Once the watershed is converted to a polygon Shapefile, a new field is added to its attribute table, and the geometry of the polygon is calculated. This step determines the area of the watershed in square miles. A search cursor is an Arcpy cursor that can read values in rows in an attribute table (Tateosian 2015). A search cursor was written in the code to retrieve the calculated drainage area within the watershed polygon attribute table. Then, just like the manually-entered value, the tool will search the reference reach database for the appropriate matches based on the pre-defined ranges.

- **Output:** All reference reach points in the database with a drainage area within the appropriate range of the manually-entered or calculated drainage area value will be used in determining the final output.

2.4.4 Valley Slope (Step 4)

- **Description**: The valley slope (ft/ft) is the difference in elevation between the starting and ending points of a stream's valley, divided by the length of that stream's valley. These values are best determined by looking at contour lines, however it can take some practice. Depending on the shape of the stream channel, the valley may curve, meaning that two or more length measurements may need to be taken and added together to get a properly estimated value. Therefore, the GUI includes three boxes in which the user must enter the two elevation values (ft) and the length (ft) in order for the tool to perform this calculation.

  Valley slope influences reach depth, and can help to explain the sinuosity and bed material of a stream. For example, downstream reaches within a watershed will tend to have a lower valley slope than those at upstream locations. Steeper streams will have lower sinuosity than flat streams, but tend to have larger bed material size (Doll et al.). Due to the valley slope's influence on several stream characteristics, it is best to have a narrow range of values (+/- 0.5 percent) that are used as a search criteria within the tool based on the user input value.

- **Rule:** Reference reach valley slope must be within +/- 0.5 percent of the valley slope of the restoration project of interest. A user-defined function was created to perform the calculations and properly carry out this rule:

```
def ValleySlopeRange(valleySlope):
    tolerance = float(valleySlope * 0.05)
    plusTolerance = float(valleySlope + tolerance)
    minusTolerance = float(valleySlope - tolerance)
    return (minusTolerance, plusTolerance)

slopeSelectionRange = ValleySlopeRange(valleySlope)
```

- **Use:** User will manually enter a numerical value for the valley slope (ft/ft) of the restoration project of interest.
- **Option:** If actual valley slope of the project reach is unknown, the user can choose to fill out the "Valley Slope Calculator" within the Reference Reach Finder Tool.
    - **Use:** The "Valley Slope Calculator" requires the user to provide a contour lines file as input. The user is asked to specify the field within the contour lines file that contains the elevation values. In the code, the default field value is set to 'Z_Feet' based on the contour lines file used for testing. The code instructs the tool to add the elevation labels to the contour lines when it is run. The following snippet shows how this was achieved in the code:

```
    # Display elevation ('Z_Feet') labels on map for user to view
contourLayer = arcpy.mapping.ListLayers(mxd, contoursAddToMap)[0]

if contourLayer.supports("LABELCLASSES"):
```

```
                for lblClass in contourLayer.labelClasses:
                    lblClass.showClassLabels = True
            lblClass.expression = "[Z_Feet]"
            contourLayer.showLabels = True
```

However, the user will need to be able to see these labels before the tool is run in order to enter in some of the values needed for the calculation. Additionally, the user will need to use the Measure tool on the contour lines of interest to estimate the third value needed for the calculation, which can only be done before opening the tool. These limitations are discussed in detail in the **Discussion and Conclusion** section of this report.

- **Output:** All reference reach points that have a valley slope within 0.5 percent of the manually-entered or calculated value will be selected and used in determining the final output.

2.4.5 Median Grain Size (D50) (Step 5)

- **Description:** Knowledge of the bed material of a stream reach is important for stream design. The median grain size (D50) is a good representation of the overall particle size of sediment within a stream. It is important for understanding how water will physically flow through a stream, how water will be absorbed into the material, and a good judge of the overall stability of a stream.
- **Rule:** Reference reaches must have the same channel material as the restoration project of interest, based on the median grain size (D50)

- **Use:** The user must select the channel material category that is associated with the pre-determined median grain size (D50 (mm)) of the restoration project. The D50 particle size range can be chosen from a drop-down menu with the options indicated in **Table 2**.

Table 2*: Particle size ranges (mm) for channel material classification in Reference Reach Finder tool.*

Particle Size Range (mm)	Channel Material
< 0.064	Silt/Clay
0.064 - < 2.0	Sand
2.0 - < 64.0	Gravel
64.0 - < 256.0	Cobble
>/= 256.0	Boulders

- **Output:** Channel material of reference reach must match the desired median grain size (D50) of the restoration project of interest.

3.0 Results

3.1 North Carolina Reference Reach Sites

The final reference reach database Shapefile contains 130 points (**Figure 4**). The associated database table containing all of the reference reach sites and their values for each parameter is in **Appendix B**. Data availability varied across mitigation plans- some reports more detailed than others. Unknown values are indicated within the table.

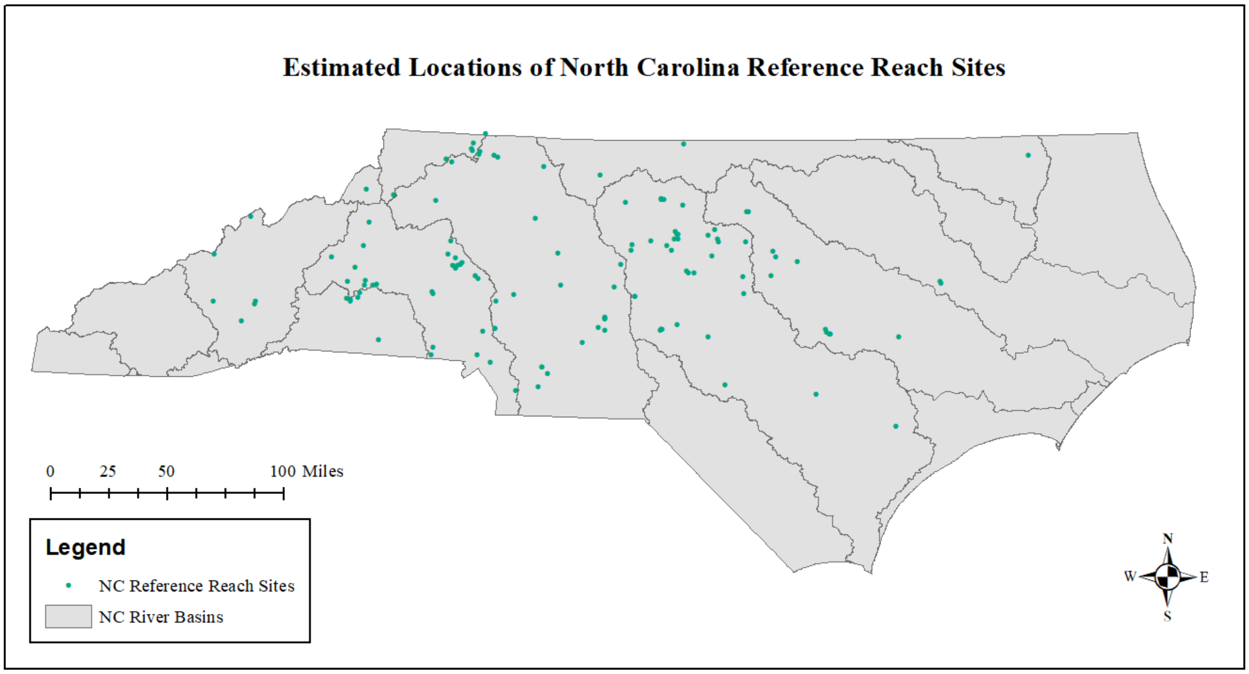

Figure 4. *Distribution of reference reach site locations in the final database.*

3.1.1 Geographic Distribution of Reference Reach Sites

Figure 4 shows a map of 130 points, with each point representing one reference reach site within the database. There appears to be a low evenness in geographic distribution among the reference reach points in the database. There are both clusters of reference reaches and large gaps. For example, six of the seventeen North Carolina river basins currently have no representation within the database. The Hiwassee, Savannah, Little Tennessee, Lumber, Pasquotank and White Oak river basins have no reference reach points within their boundaries. The Catawba, Yadkin Pee Dee, Cape Fear and Neuse river basins have the highest number of reference reach sites located within their boundaries. The eastern part of the state, especially near the coast, is devoid of sites. Additionally, the most western part of the state also lacks reference reach locations. The middle part of the state, also called the Piedmont region, contains the most points within this database.

Figure 5 shows a map of all one hundred North Carolina counties, and uses a graduated color scale to portray the number of reference reach sites per county in the database. The aforementioned geographic distribution gaps can be seen very clearly on this map. Fifty-seven of the one hundred counties are not represented in the database, and are therefore displayed with the lightest color on the map. Of the forty-three represented counties, the average number of reference reach sites is 3. Seventeen counties only have one reference reach within their boundaries. Six counties have two sites, three counties have three sites, five counties have four sites, six counties have 5 sites, three counties have six sites, one county has eight sites, and one county has ten sites (**Table 3**). Alamance and Catawba counties have the largest number at ten and eight reference reach sites, respectively, within their boundaries. These, and other counties with high site counts, are located close to large urban centers like Charlotte, Greensboro, Durham, Chapel Hill and Raleigh. Western and eastern North Carolina counties are underrepresented in the database, with most having zero or only one site within their boundaries. As the mitigation industry in North Carolina grows, the geographic range of reference reach sites within the state will likely expand as well.

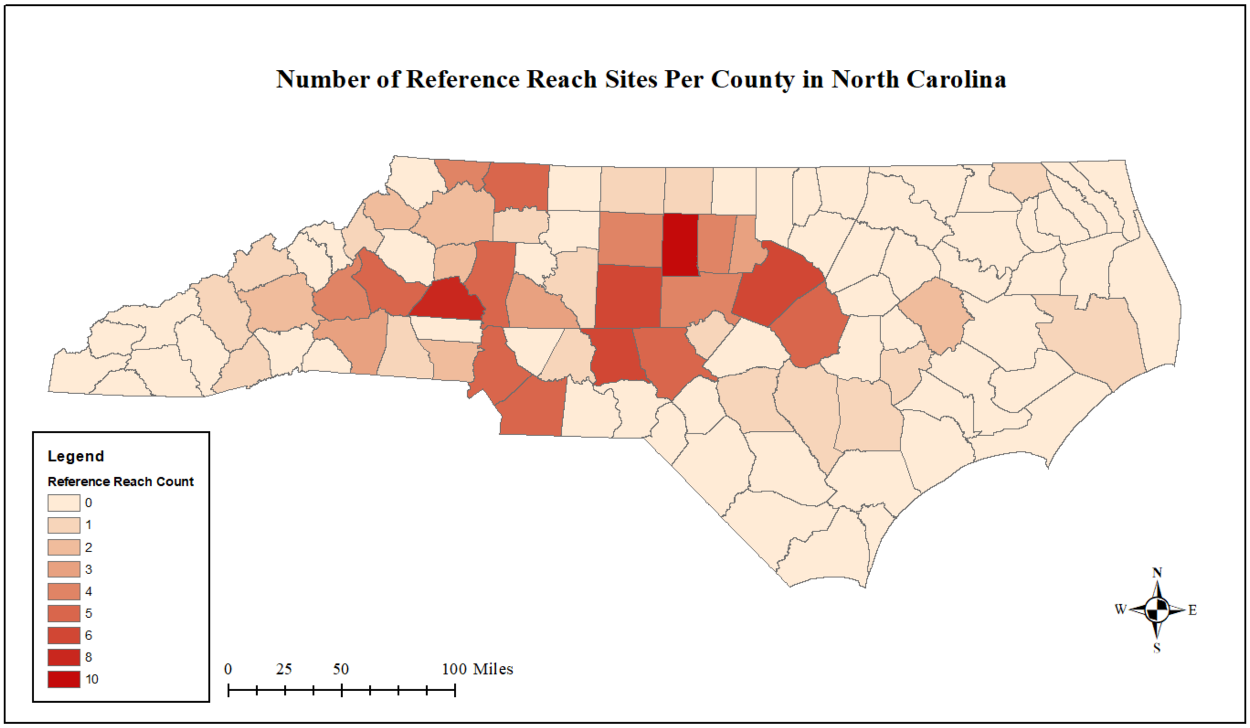

Figure 5. *Number of reference reach sites per county in North Carolina. Numbers based on the results of reference reach database presented in this report.*

Table 3. *Number of reference reach sites in the North Carolina counties that are represented within the reference reach database presented in this report.*

NC County Name	Number of Reference Reach Sites
Alamance	10
Alexander	2
Alleghany	4
Avery	1
Buncombe	2
Burke	5
Caswell	1
Catawba	8
Chatham	4
Cleveland	1
Cumberland	1
Davidson	1
Duplin	1
Durham	3
Gaston	2
Gates	1
Guilford	4
Haywood	1
Hyde	1
Iredell	5
Johnston	5
Lee	1
Lenoir	1
Madison	1
McDowell	4
Mecklenburg	5
Montgomery	6
Moore	5
Orange	4
Pitt	2
Randolph	6
Rockingham	1
Rowan	3
Rutherford	3
Sampson	1
Stanly	1
Surry	5
Transylvania	1
Union	5
Wake	6
Watauga	2
Wilkes	2
Yadkin	1

3.1.2 Reference Reach Database Parametric Results

Within the reference reach database created for this study, only four of the six stream types included in the tool are present (**Table 4**). Stream types G and F appear zero times within the database. Only four reference reaches in the database are type A, twenty-nine as stream type B, forty-nine are classified as stream type C, and stream type E had the highest representation in the database with sixty results. There was a slight amount of overlap within the results due to some mitigation plans listing two stream types per reference reach. This likely occurred if the mitigation plan was referring to two different reaches, but classifying them together as one reference reach because they were on the same stream channel. Because reference reaches are considered stable streams that can serve as models for restoration design, it is to be expected that there are zero or very few that are classified as A, G or F stream types. These stream types fall under the "entrenched" category in the Rosgen Classification of Natural Rivers. An entrenched stream flows through a narrow, often deep valley, and it is characterized by extremely eroded banks. The erosion can be caused by flooding and other consequences of urbanization. Therefore, the greatest number of reference reaches in the database are classified as the stream types that are considered to be more stable and less degraded (B, E, and C).

Table 4. *Number of reference reaches within the NC database presented in this report that belong to each Rosgen Stream Type letter for single thread channels.*

Rosgen Stream Type Letter	Number of Reference Reaches in Database
A	4
G	0
F	0
B	29
E	60
C	49

The database includes reference reaches with a wide range of drainage area (mi^2) values (**Table 5**). The minimum drainage area is 0.014 mi^2, and the maximum is 24.80 mi^2. The median and median absolute deviation values are 0.5 mi^2 and 0.2 mi^2, respectively. Therefore, the maximum drainage area value was an outlier, and most values within the database remained on the lower end of the range. The drainage area ranges (**Table 1**), which are used as search criteria within the tool, are primarily based on these results. The tool assumes that the input drainage area will be less than thirty square miles, but users can search for reference reaches with drainage areas of up to 34.99 square miles. If needed, the code could be easily amended to include a higher range of input values. However, according to the results of this large sample database, that was deemed unnecessary. The searchable ranges were somewhat arbitrarily determined based on the input value. Smaller drainage area values will tell the tool to search for references reaches within a narrower range of values than larger drainage area values, and vice versa.

Table 5. *Drainage area (mi^2) statistical results from the NC reference reach database presented in this report.*

Reference Reach Database: Drainage Area (mi^2) Statistics	
Minimum	0.014
Maximum	24.80
Median	0.5
Median Absolute Deviation	0.2

The valley slope (ft/ft) results yielded a smaller range of values than the drainage area results, which was to be expected (**Table 6**). Within the database, the minimum valley slope is 0.001 ft/ft, and the maximum is 0.1176 ft/ft. The median and median absolute deviation values are 0.012 ft/ft and 0.002 ft/ft, respectively. Because the valley slope ratios are small numbers, a criteria of +/- 0.5% was implemented in the tool as the searchable range within the reference reach database. For example, if the user were to enter the average valley slope value (0.0166 ft/ft) as input, the tool would search for reference reaches with valley slope values in the range of 0.0083 and 0.0249 ft/ft.

Table 6. *Valley slope (ft/ft) statistical results from the NC reference reach database presented in this report.*

Reference Reach Database: Valley Slope (ft/ft) Statistics	
Minimum	0.001
Maximum	0.1176
Median	0.012
Median Absolute Deviation	0.002

Lastly, the median grain size (D50) results were much like those of the stream type classification, meaning that two of the five categories were listed for none of the reference reaches (**Table 7**). The silt/clay (<0.064 mm) and boulder (>= 256.0 mm) particle sizes had zero reference reach results. The cobble (64.0 mm – 256.0 mm) classification was close behind the former two with only six reference reaches. Twenty-four reference reaches are classified as sand (0.064 mm - < 2.0 mm), and gravel (2.0 mm – 64.0 mm) has the greatest number of D50 classifications at sixty. It is to be expected that all of the reference reaches in the database fall within the three middle particle size ranges for D50. If the median particle size of a stream's sediment material was too fine, like silt or clay, or too heavy, like big cobble and boulders, the channel would be unstable. A stable stream channel should be able to transport some of its sediment and other materials downstream, but it should also be able to maintain its natural structure by holding onto materials in the streambed and along the banks.

Table 7. *Number of reference reaches within the NC database presented within this report that belong to each median grain size (D50) category.*

D50 Classification	Number of Reference Reaches in Database
<0.064 mm (silt/clay)	0
0.064 mm - < 2.0 mm (sand)	24
2.0 mm - < 64.0 mm (gravel)	61
64.0 mm - < 256.0 mm (cobble)	6
>= 256.0 mm (boulders)	0

3.2 Graphical User Interface

The graphical user interface (GUI) for the Reference Reach Finder Tool Setup tool is shown in **Figure 6**. The GUI for the main tool uses a categorized design, which is useful when many parameters are involved. ArcMap does not allow the option to create tools using “Next” buttons when many steps are involved, for example, so the categorized, collapsible design is the best way to achieve organization of parameters. **Figure 7** shows the GUI when all steps are collapsed. The workspace, output directory, and the reference reach points input file are all displayed at the top outside of the categorized design because they relate to the initial setup of the tool rather than a specific parameter category. **Figures 7, 8, 9, 10** and **11** show the expanded GUI designs for each of the main parameters. To increase flexibility for the user, each parameter has been set to optional. Default values are set for several input files and field names. For example, the NC Reference Reach Points Shapefile is set as the default input file, and field names within that file are set as default values for each main parameter (‘Name’, ‘ST_Letter’, ‘Drainage_A’, ‘V_Slope’, ‘D50_Class’).

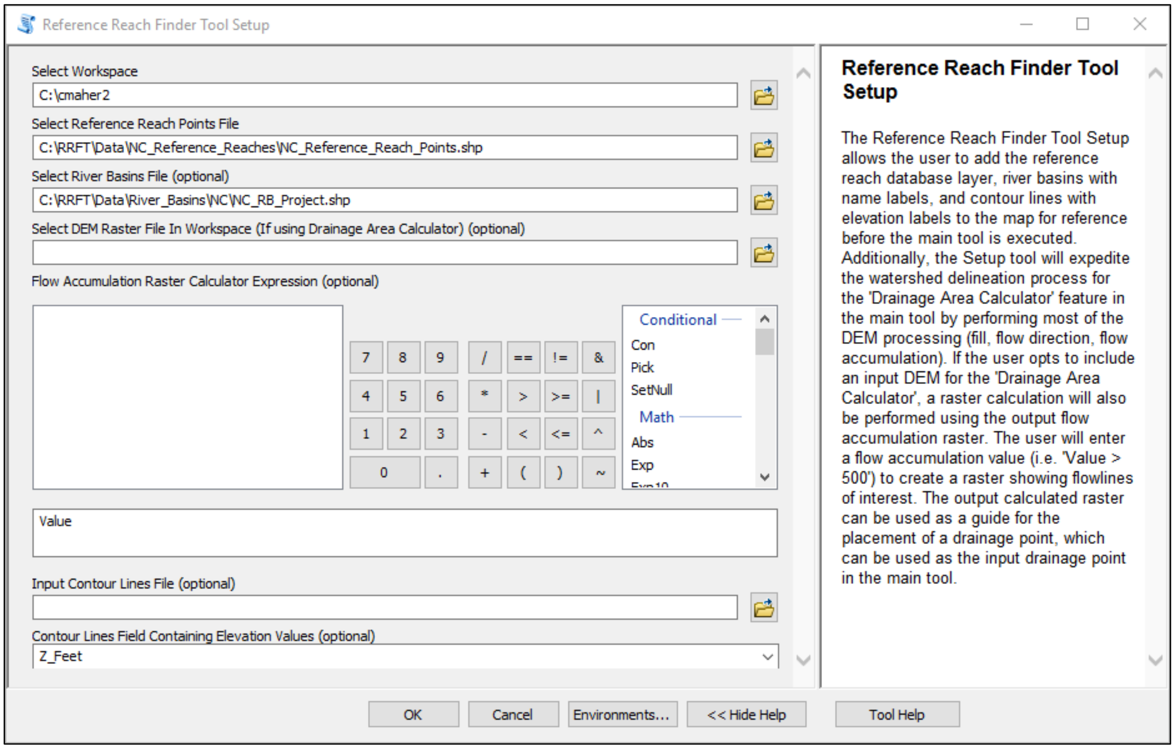

Figure 6. *Refererence Reach Finder Tool Setup (RRFT-S) tool GUI.*

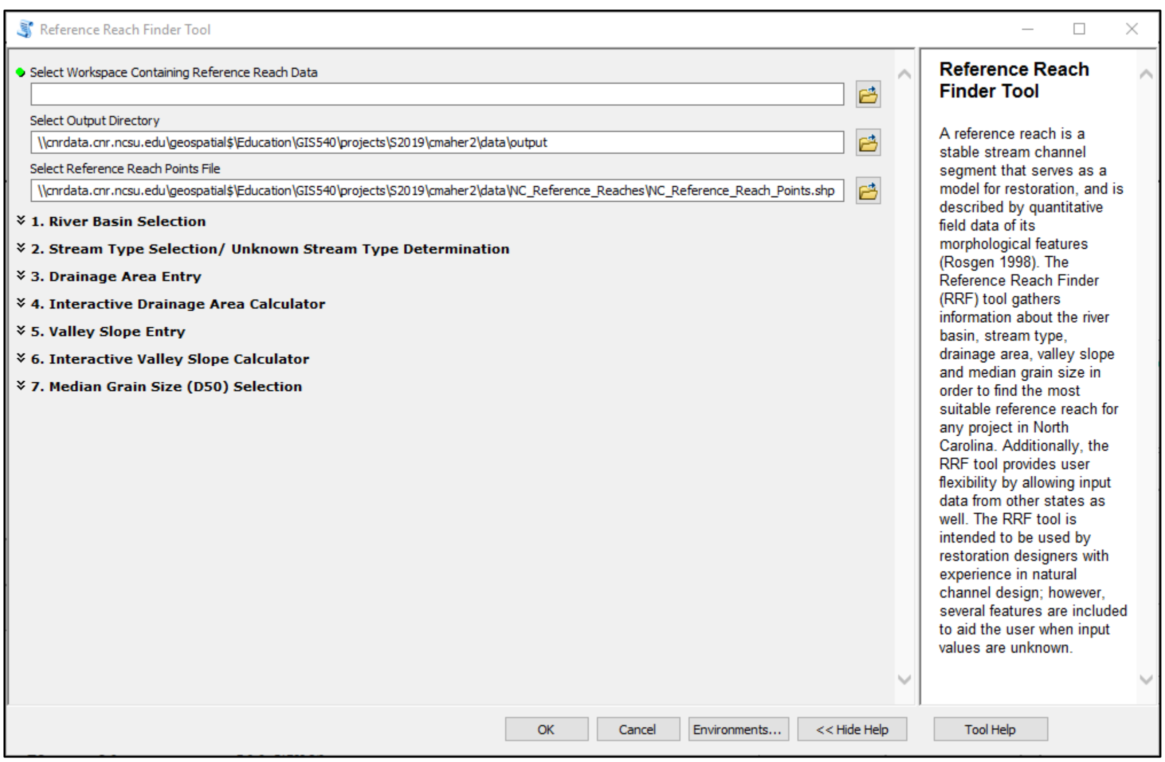

Figure 7. *Collapsed categorized script tool GUI design for the Reference Reach Finder Tool.*

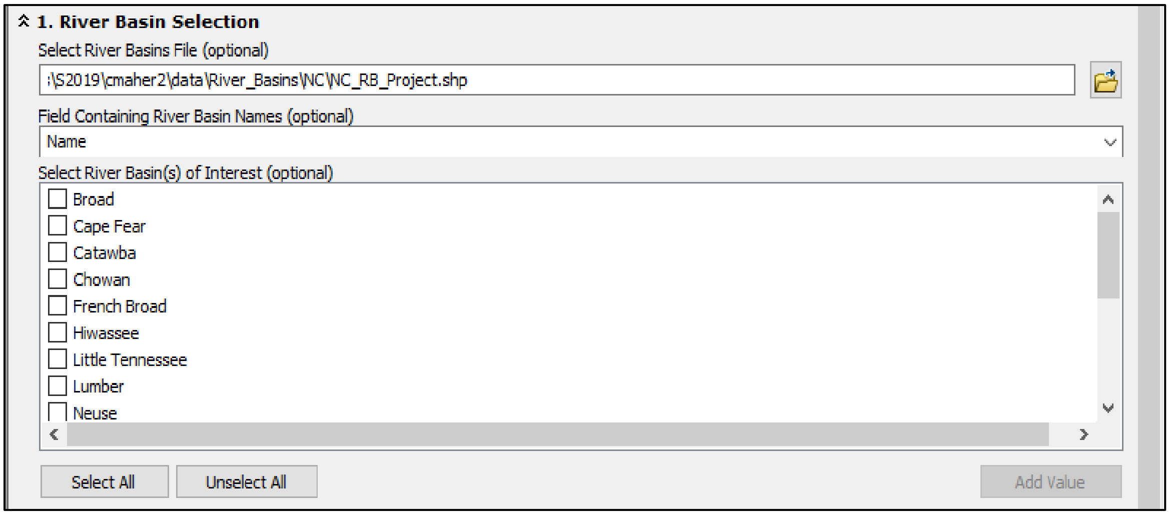

Figure 8. *River basin selection GUI design for the Reference Reach Finder Tool.*

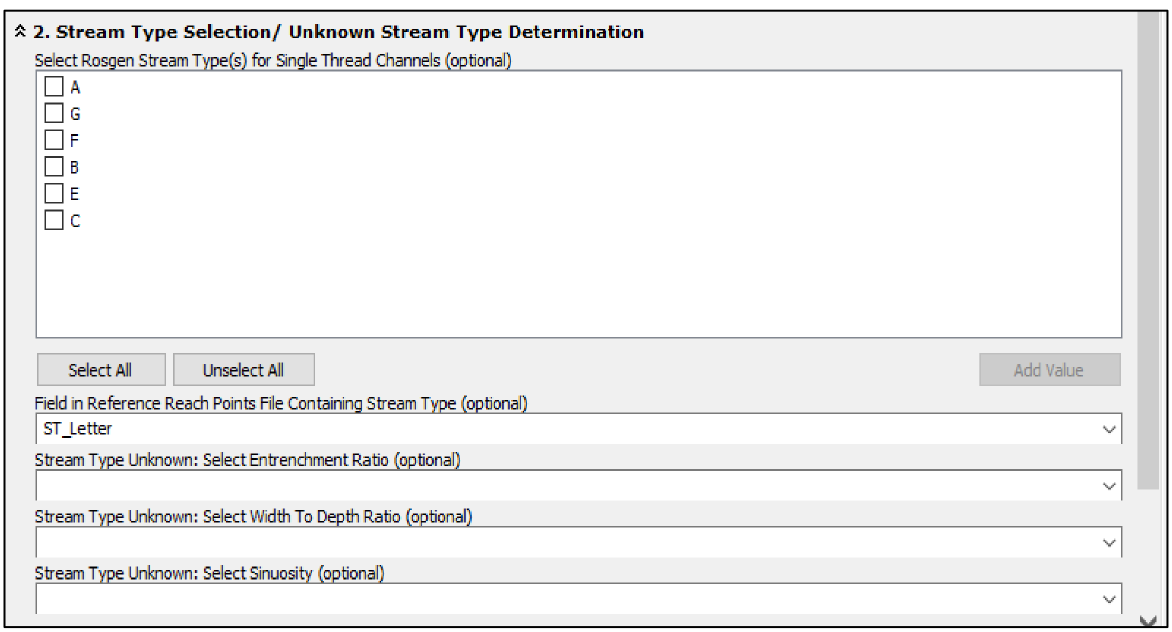

Figure 9. *Rosgen Stream Type selection GUI design for the Reference Reach Finder Tool.*

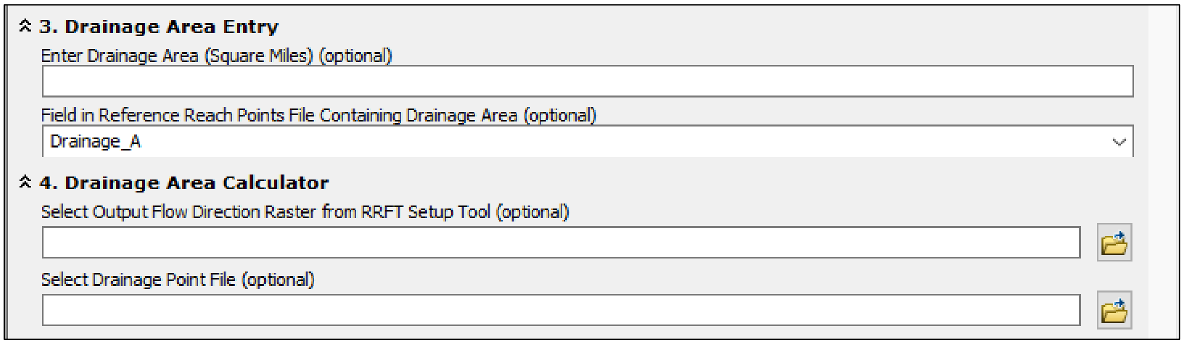

Figure 10. *Drainage Area entry and drainage area calculator GUI design for the Reference Reach Finder Tool.*

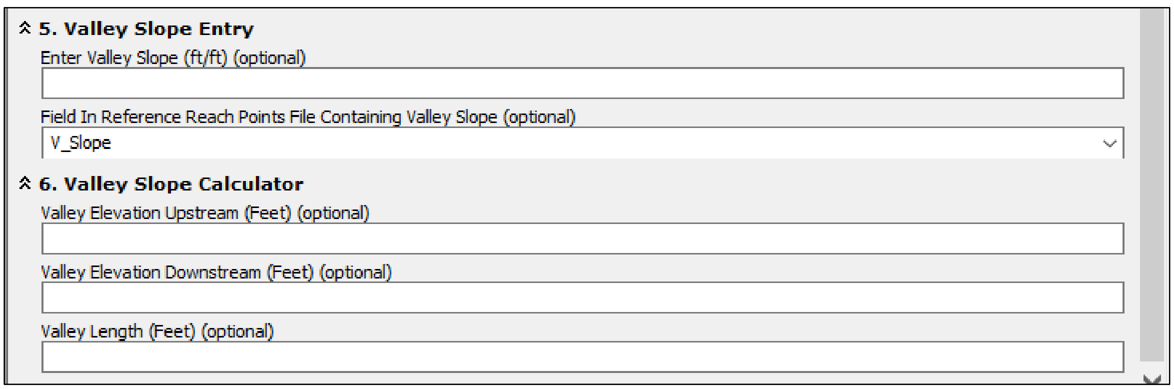

Figure 11. *Valley slope entry and valley slope calculator GUI design for the Reference Reach Finder Tool.*

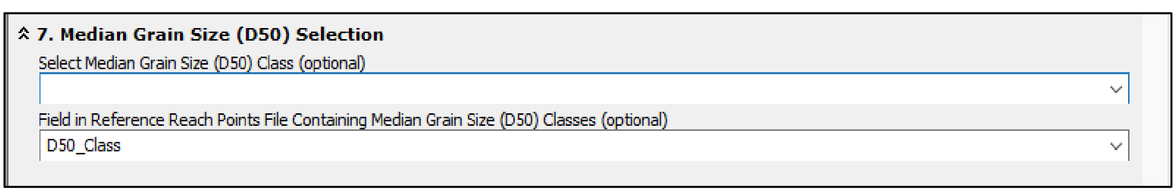

Figure 12. *Median Grain Size (D50) selection GUI design for the Reference Reach Finder Tool.*

4.0 Discussion and Conclusion

4.1 Assessment of Reference Reach Site Locations

Due to data limitations, the majority of the sites are estimated to the closest possible location. In order to obtain more precise data, geographic coordinates would need to be recorded in the field. However, this database is accurate enough to be used a means for evaluating the potential need for new reference reach sites across the state.

It is likely that most reference reach points are concentrated within the Piedmont region because that is where the most urban development occurs within the state. The more urban development that occurs, the higher the demand for stream restoration, and therefore, a higher demand for reference reaches. Morphologically, it would not make much sense to choose a reference reach site in the mountains or near the coast for a restoration project that will be constructed in the Piedmont region. For example, the topography in the mountains is much different than that in Raleigh, NC, and therefore, the streams have different morphological dimensions and structures. The mitigation industry in North Carolina is primarily based in large urban centers in the Piedmont region, so this could be another reason why there are large geographic gaps in the reference reach database. There have been many recent cases of illegal dumping and hazardous chemicals entering waterways near the coast. Additionally, the recent hurricanes that have made landfall on the North Carolina coast, Hurricane Matthew (2016) and Hurricane Florence (2018), have caused pollution and structural degradation to streams and wetlands in coastal areas. Based on these recent events, it seems that there is an increasing need for stream and wetland restoration near the North Carolina coast. In order to properly restore waterways near the coast, using natural channel design methods, finding appropriate reference reach sites is crucial. The six river basins that are not represented in the reference reach database should be prioritized for future database expansion, especially those in coastal regions.

4.1 Graphical User Interface Limitations

The original plans for the GUI were made before the extent of the capabilities of a Python Toolbox GUI were known. Generally, it was assumed that the GUI would allow the user to select points or measure lines on the map and use those features as input in the tool. However, there is little information available on the ArcGIS Resources website or other online resources about this specific functionality, and it was unable to be implemented into the tool due to time constraints. Allowing the user to interact with the map while using the tool is likely more easily implemented on ArcGIS Online web maps and web apps, rather than ArcGIS Desktop software. Implementing the original plans into an online Reference Reach Finder Tool could be a good area for future research.

The original plans for the “Interactive Drainage Area Calculator” were affected by the aforementioned GUI limitations. Originally, the user would be able to select a point on the map and use it as an input drainage point while using the tool. Now, the user must select an existing drainage point file to use this feature. To account for the change in plans, a tutorial video was created showing the user how to create a point feature class for their drainage point of interest. If the user was able to select a drainage point on the map, it would be helpful if they could use the raster calculated flow accumulation DEM. This DEM shows flowlines that could serve as a guide for where the user may choose to place a drainage point. However, layers generated by the tool cannot be added to the map until the tool is run. This issue was resolved by the creation of the Reference Reach Finder Tool Setup (RRFT-S) tool. If the user wants to use the raster calculated flow accumulation DEM as a guide for drainage point placement, they would be required to run the RRFT-S tool with their DEM file and enter a flow accumulation query. Then, they will need to create a drainage point feature class, place their point on a flowline of interest, and then use that new drainage point Shapefile as input in the main tool for completing the watershed delineation.

The original plans for the 'Valley Slope Calculator' were also hindered by GUI limitations. Originally, the tool added the input contour lines file with elevation labels to the map after the tool is run, which kept the user from extracting elevation values for the valley slope calculation while using the tool. Therefore, the upstream and downstream elevation values must be known prior to using the main tool, or else the 'Valley Slope Calculator' is a pointless tool feature. Additionally, the Measure tool in ArcMap can be used to measure the length of the valley of the stream, using the contour lines as a guide. However, this must be done before the main tool is used, and cannot be used when other tools are in use. This problem is also solved by the RRFT-S by adding the contour lines with elevation labels to the map when executed.

4.3 Anticipated Impacts

As previously mentioned, mitigation providers generally only have a small selection of reference reach sites to choose from for each project. It is likely that, if time permitted, providers would prioritize expanding their selection of reference reach sites by planning more field surveys. However, the current demand for mitigation credits is high due to steady increases in urban development, so there is a constant influx of new projects and little time for tasks like reference reach site surveys. The mitigation industry can be competitive among private local companies, where much of their work is kept internally confidential. However, mitigation reports, among other documents, are often published online through the state government division that is responsible for regulating mitigation projects. Therefore, the small reference reach selection of several mitigation companies was available to be consolidated into a large database. A reference reach database has not been created using the same methods or to the same extent and detail as the one included in this project. By providing North Carolina restoration designers with a much larger pool of reference reach sites, they are likely to find better restoration models for upcoming projects, with morphological features that are a better match than those from a smaller selection of sites. Because the reference reach database created for this project is very large at 130 sites, a primary goal of the tool was to create a means for better spatial analysis and easier attribute selection, when compared to looking through a data table. The RRFT provides the user with an interactive and organized way to explore the database, as well as visualize the distribution of the current selection and assess the need for future field surveys.

In order to restore the biological and physical functions of an unstable stream channel, the appropriate morphological dimensions for that stream must be implemented in the design criteria. Over twenty years ago, Rosgen published the concept of the reference reach, and he explains that the integrative approach of this method can keep streams from aggrading or degrading (Rosgen 1998). Today, the Reference Reach Finder Tool modernizes this approach by maximizing the database and optimizing the results. Better reference reaches can produce more successful mitigation projects and ecological benefits. For example, vertical channel stability is a critical component of a successful mitigation project. When a stream has vertical channel stability, it does not have degradation or aggradation. Rather, it is in a dynamic equilibrium between transporting and receiving bed material. In order to achieve vertical stability, stream designers may need to observe the dimensions, valley type, bed material and geologic region of multiple reference reaches to tailor an appropriate design criteria for the project. This reference reach assessment can easily be made by using the RRFT. It would be more difficult for restoration designers to figure out the correct dimensions and insert the appropriate in-stream structures to achieve vertical stability without access to the tool's database and features.

There could be major ecological consequences for a restoration project if vertical instability is a factor. For example, degradation can occur when the stream is out of equilibrium by transporting more bed material than is received, causing entrenchment. Entrenchment makes it more difficult for a stream to reach the bankfull stage and saturate its floodplains, which creates a positive feedback loop by increasing the transport of materials through the channel. A lack of natural periodic saturation could adversely affect the biota of the floodplains, especially the vegetated buffer. If the trees and other plants growing in the buffer zone die, stability of the streambanks could further decrease and cause more erosion to occur. When a stream is devoid of a vegetated buffer, which is a natural filtration system, more pollution is likely to enter the channel through runoff. An increase in pollutants can harm the aquatic life, potentially making the stream inhabitable to macroinvertebrates and fish. This ecological chain reaction is just one example of why using the appropriate morphological dimensions for natural channel design are critical for successful restoration. Modernizing Rosgen's reference reach approach with the RRFT is necessary to create quality restoration designs and ultimately meet the growing demand for mitigation.

4.4 Conclusion

The primary objective of this project was to provide stream restoration designers and other professionals in the field with an easier way to find optimal reference reach sites through the use of a GIS-based tool. The Reference Reach Finder Tool provides a comprehensive method of selecting sites by including the key parameters that are used to classify whether or not a reference reach can serve as a guide for the project of interest. In North Carolina, specifically, the reference reach database created for this tool greatly expands the reference reach site options that are available to stream restoration designers in the state. Beyond the goals for the RRFT, several other academic and professional objectives were met. Throughout this project, I was able to increase my knowledge of stream restoration design practices and principles, and learn new technical skills, like coding in Python, to increase my marketability to employers and literacy within the field. Ultimately, knowledge of Python coding and creating geospatial tools can be applied to other areas of natural resources and hydrology to improve efficiency and solve problems.

There are some improvements that could be made, such as expansion of the database in underrepresented river basins and creation of an additional tool to allow the user to collect preliminary data before the main tool is executed. The tool package includes data specific to North Carolina, and therefore it would be easiest to execute locally, but it can be used out-of-state as well. The tool provides user flexibility in several ways including the option to use input data from outside of North Carolina, setting all parameters as optional steps and providing alternative methods when values are unknown. Additionally, tool parameters may be adjusted or added. For example, some experts in the field might rather have ecoregions or other types of ecological boundaries rather than the watershed boundaries when filtering reference reach sites. Parameters that should be considered as future additions to the tool are percentage of land use/land cover (LULC) within the watersheds of reference reaches, and the biological health of reference reach sites and connecting streams. Finally, the coding of the tool can be improved by implementing a better parameter filtering criteria system as well as the “interactive” features that were originally planned for the GUI.

The tool provides a means for spatial analysis and assessment of the reference reach database. Additionally, the tool makes it easier to navigate the database and produce results based on the parameters that are of highest importance to the user. Using a graphical user interface to select parameter properties and executing the tool is an easier and more organized method of finding appropriate reference reach sites than scrolling through a large database table. The efficiency and quality of the stream restoration design process is anticipated to improve with the use of the RRFT. With a wider selection of reference reach sites, it is likely that a particular restoration project will be paired with one or multiple reference channels that serve as a better model for design. Several reference reaches with similar or different morphological criteria may be consulted to tailor an optimal design plan, resulting in a restored stream that maximizes its ecological potential. Finally, the RRFT provides a visual geospatial representation of the reference reach database, which allows for assessment of overserved or underserved regions. As urban development continues to increase across the United States, the demand for mitigation credits will also increase. It is anticipated that this rise in demand will naturally result in an expanded distribution of surveyed reference reach sites across North Carolina and the nation. The Reference Reach Finder Tool improves the process of selecting the appropriate morphological dimensions that are necessary for the physical and biological functions of a stream, and ultimately optimizes ecological potential by aiding to produce better quality restoration designs.

6.0 References

Allan, J. David., and M. María Castillo. *Stream Ecology: Structure and Function of Running Waters*. Springer, 2009.

Doll, Barbara A., et al. "Stream Restoration: a Natural Channel Design Handbook." North Carolina Stream Restoration Institute; North Carolina Sea Grant.

Georgakakos, A., et al. *Ch. 3: Water Resources. Climate Change Impacts in the United States: The Third National Climate Assessment*, J. M. Melillo, Terese (T.C.) Richmond, and G. W. Yohe, Eds., U.S. Global Change Research Program, 69-112. 2014.

Hauer, F. Richard., and Gary Anthony. Lamberti. *Methods in Stream Ecology*. 2nd ed., Academic Press/Elsevier, 2006.

Horner, Richard, et al. "Structural and Non-Structural BMPs for Protecting Streams." *Linking Stormwater BMP Designs and Performance to Receiving Water Impact Mitigation*, 2002, doi:10.1061/40602(263)6.

Kannan, Narayanan, et al. "Estimation of Stream Health Using Flow-Based Indices." *Hydrology*, vol. 5, no. 1, 2018, p. 20., doi:10.3390/hydrology5010020.

Konrad, C P, and D B Booth. "Hydrologic Changes in Urban Streams and Their Ecological Significance." *American Fisheries Society Symposium*, no. 47, 2005, pp. 157–177.

Lenat, David R., and J. Kent Crawford. "Effects of Land Use on Water Quality and Aquatic Biota of Three North Carolina Piedmont Streams." *Hydrobiologia*, vol. 294, no. 3, Dec. 1994, pp. 185–199., doi:10.1007/bf00021291.

Meyer. "Small Streams and Wetlands Provide Beneficial Ecosystem Services." *University of Montana*, Sierra Club, American Rivers , 2003, hs.umt.edu/dbs/labs/lowe/documents/teaching/Readings/Meyer_et_al_2003.pdf.

Palmer, Margaret A., et al. “Ecological Restoration of Streams and Rivers: Shifting Strategies and Shifting Goals.” *Annual Review of Ecology, Evolution, and Systematics*, vol. 45, no. 1, 2014, pp. 247–269., doi:10.1146/annurev-ecolsys-120213-091935.

Palmer, Margaret A., et al. “From Ecosystems to Ecosystem Services: Stream Restoration as Ecological Engineering.” *Ecological Engineering*, vol. 65, Apr. 2014, pp. 62–70., doi:10.1016/j.ecoleng.2013.07.059.

Rosgen, Dave. “The Reference Reach: A Blueprint for Natural Channel Design.” *Engineering Approaches to Ecosystem Restoration*, 1998, doi:10.1061/40382(1998)166.

Rosgen, David L. and Pacific Wildland. “A STREAM CHANNEL STABILITY ASSESSMENT METHODOLOGY BY.” (2001).

Schoonover, Jon E., et al. “Impacts of Land Cover on Stream Hydrology in the West Georgia Piedmont, USA.” *Journal of Environment Quality*, vol. 35, no. 6, Nov. 2006, pp. 2123–2131., doi:10.2134/jeq2006.0113.

Stoddard, John L., et al. “Setting Expectations For The Ecological Condition Of Streams: The Concept Of Reference Condition.” *Ecological Applications*, vol. 16, no. 4, 1 Aug. 2006, pp. 1267–1276., doi:10.1890/1051-0761(2006)016[1267:seftec]2.0.co;2.

“Stream Ecology.” *Stream Ecology*, Smithsonian Environmental Research Center, serc.si.edu/research/research-topics/biodiversity-conservation/stream-ecology.

“Stream Mitigation Guidelines.” North Carolina Department of Transportation, Apr. 2003.

Sudduth, Elizabeth B., et al. “Stream Restoration Practices in the Southeastern United States.” *Restoration Ecology*, vol. 15, no. 3, 26 July 2007, pp. 573–583., doi:10.1111/j.1526-100x.2007.00252.x.

Tateosian, Laura. *Python for ArcGIS*. Springer, 2015.

Walsh, Christopher J, et al. *Urban Stormwater and the Ecology of Streams* . Cooperative Research Center for Freshwater Hydrology, Cooperative Research Center for Catchment Hydrology, urbanstreams.net/lsc/Docs/UrbanStormwaterTechreport.pdf.

White, Kirk E. "Regional Curve Development and Selection of a Reference Reach in the Non-Urban, Lowland Sections of the Piedmont Physiographic Province, Pennsylvania and Maryland." *U.S. Geological Survey*, 2001, doi:10.3133/wri014146.

Yeakley, J. Alan, et al. "Ecosystem Services of Streams and Rivers." *River Science*, 2016, pp. 335–352., doi:10.1002/9781118643525.ch17.

Yochum, Steven E. "Guidance for Stream Restoration." *United States Department of Agriculture* , Forest Service: National Stream and Aquatic Ecology Center, May 2018, www.fs.fed.us/biology/nsaec/assets/yochumusfs-nsaec-tn102-4guidancestreamrestoration.pdf.

Appendix A.

Reference Reach Sites Used in NC DMS Mitigation Plans 2010-2018

KEY:

- **Site Name**
  - Location of geomorphic parameters data

  Page numbers are based on Microsoft Edge/ Adobe viewers and NOT page numbers listed on mitigation plan reports
- Mitigation plan does NOT provide sufficient geomorphic parameters data OR general location cannot be determined.

REFERENCE REACHES:

- **Cedarock**
  - (Abby Lamm MP pages 18-19)
  - (Aycock Springs MP pages 19-20)
- **Causey Farm**
  - (Abby Lamm MP pages 18-19)
- **UT to Richardson Creek**
  - (601 East MP pages 88-89, 94-95)
- **UT to Rays Fork**
  - (601 North MP page 178)
- **Brookhaven**
  - (Abbot MP (1999) pages 36-37)
- **Mingo Creek**
  - (Abbot MP (1999) pages 36-37)
- **Spencer Creek**
  - (Scaly Bark Creek MP pages 25-26)
  - (Town Creek page 184)
  - (Upper Silver Creek MP pages 323 and 324)
  - (UT to Town Creek pages 50-52)
  - (Agony Acres MP pages 29-30)
  - (Bear Creek MP pages 123-125)
  - (Big Harris Creek pages 113-114)
  - (Byrd’s Creek MP page 28)
  - (Candy Creek MP pages 33-37)
  - (Crooked Creek #2 MP pages 42-43)
  - (Deep Meadow MP pages 145-148)
  - (Foust Creek MP pages 30-32)
  - (Little Troublesome Creek MP pages 31-32)

- (Lochill Farm MP pages 25-28)
- (Martin Dairy MP page 18)
- (Mud Lick Creek MP pages 22-25)
- (Norkett Branch MP pages 44-46)

- **UT to Spencer Creek**
  - (Norkett Branch MP pages 44-46)
- **UT to Polecat Creek**
  - (Agony Acres MP pages 29-30)
  - (Holman Mill MP pages 24-25)
  - (Martin Dairy MP page 19)
  - (Mud Lick Creek MP pages 22-23)
- **UT(s) to Cane Creek**
  - (Agony Acres MP pages 29-30)
  - (Deep Meadow MP pages 145-148)
  - (Underwood pages 37-38)
  - (Byrd's Creek MP page 28)
  - (Foust Creek MP pages 32-34)
  - (Maney Farm MP pages 27-28)
  - (Mud Lick Creek MP pages 23-25)
  - (Big Harris Creek MP 109-110)
  - (Deep Meadow MP pages 145-148)
- **Agony Acres UT1 Reach 3**
  - (Agony Acres MP pages 29-30)
  - (Candy Creek MP pages 35-37)
  - (Martin Dairy MP page 19)
- **Cripple Creek**
  - (Aycock Springs MP pages 19-20)
- **Group Camp Tributary**
  - (Vile Creek MP pages 56-58)
  - (Big Harris Creek MP pages 109-110)
  - (Henry Fork MP pages 49-50)
- **Boyd Branch**
  - (Big Harris Creek MP pages 109-110)
- **UT to South Crowders**
  - (Big Harris Creek MP pages 109-110)
  - (Lone Hickory MP pages 43, 44, 46)
  - (Henry Fork MP pages 49-50)
- **Box Creek**
  - (Big Harris Creek MP pages 113-114)
- **Hall Creek**
  - (Big Harris Creek MP pages 113-114)
- **Meadow Fork**

- (Big Harris Creek MP pages 113-114)
- (Little Pine Creek II MP pages 12-13)
- (Little Pine Creek III MP page 35)

- **UT to Gap Branch**
  - (Vile Creek MP pages 56-58)
  - (Big Harris Creek MP pages 118-119)
  - (Henry Fork MP pages 49-50)
- **UT to Kelly Branch**
  - (Big Harris Creek MP pages 118-119)
  - (Lone Hickory MP pages 42, 200)
- **UT to Sandy Run**
  - (Big Harris Creek MP pages 118-119)
- **UT to Rocky Creek**
  - (Scaly Bark Creek MP pages 19 and 20)
  - (Town Creek page 184)
  - (UT to town Creek pages 50-52)
  - (Big Harris Creek MP pages 118-119)
  - (Byrd's Creek MP page 29)
  - (Candy Creek MP pages 33-35)
  - (Deep Meadow MP pages 149-150)
  - (Little Troublesome Creek MP pages 31-32)
- **UT to Reedy Fork**
  - (Brown's Summit MP page 213)
- **Buckhorn Creek**
  - (Brown's Summit MP page 213)
- **UT to Richland Creek**
  - (Byrd's Creek MP page 29)
  - (Candy Creek MP pages 35-37)
  - (Deep Meadow MP pages 147-148)
  - (Foust Creek MP pages 32-34)
  - (Norkett Branch MP pages 44-46)
- **Richland Creek**
  - (Town Creek MP page 184)
  - (UT to Town Creek MP pages 50-52)
  - (Lochill Farm MP pages 25-28)
- **Collins Creek**
  - (Candy Creek MP pages 33-35)
  - (Little Troublesome Creek MP pages 31-32)
- **UT to Varnals Creek**
  - (UT to Cane Creek MP page 156)
  - (Candy Creek MP pages 35-37)
  - (Holman Mill MP pages 24-25)
  - (Maney Farm MP pages 27-28)

    - (Martin Dairy MP page 19)
- **UT to Fisher River**
    - (Cedar Branch MP page 179)
    - (Jacobs Ladder MP page 42)
- **UT to UT1 of Cedar Creek Project**
    - (Cedar Creek MP page 275)
- **South Fork Mills River**
    - (Cochran Branch MP pages 145-147)
    - (Fletcher MP page 48)
- **Cold Springs Creek**
    - (Cochran Branch MP pages 152-153)
    - (Fletcher MP page 48)
    - (Junes Branch MP pages 140-141; 156-157; 166-167)
    - (Middle South Muddy Creek 106-107)
- **UT to Lyle Creek**
    - (Crooked Creek #2 MP pages 42-43)
    - (Lone Hickory MP pages 45 and 47)
    - (Lyle Creek MP pages 33-35)
    - (Owl's Den MP pages 48-50)
    - (Henry Fork MP pages 47-48)
- **Johanna Creek**
    - (Devil's Racetrack MP pages 47-49)
- **Bent Creek**
    - (Fletcher MP page 48)
- **Dutchman's Creek**
    - (Foust Creek MP pages 32-34)
- **UT to Catawba River (multiple sites with possibly different reaches)**
    - (Herman Dairy MP pages 20-21)
    - (Lyle Creek MP pages 33-35)
    - (Owl's Den MP pages 48-50)
    - (Henry Fork MP pages 47-48)
- **Mill Creek**
    - (Hogan Creek MP page 60)
    - (Moores Fork MP pages 85-86)
- **Agony Acres UT1A Reach 1**
    - (Holman Mill MP pages 24-25)
    - (Maney Farm MP pages 27-28)
- **UT to Bennett's Creek**
    - (Hudson MP page 91)
- **UT to Irish Buffalo Creek**
    - (Jacobs Ladder MP page 41)
- **UT to Wilkinson Creek**

- (Jacobs Ladder MP page 43)

- **Henry Fork Creek**
  - (Vile Creek MP pages 56-58)
  - (John Deere MP pages 128-130)
- **Upstream UT1 to Henry Fork**
  - (Henry Fork MP pages 49-50)
- **Little Pine Creek III UT2A**
  - (Vile Creek MP pages 56-58)
  - (Little Pine Creek II MP pages 12-13)
  - (Little Pine Creek III MP page 35)
- **UT to Belews Creek**
  - (Little Troublesome Creek MP pages 31-32)
- **Morgan Branch/ Creek**
  - (Upper Silver Creek MP pages 323-324)
  - (UT to Town Creek MP pages 50-52)
  - (Lochill Farm MP pages 25-28)
- **UT to South Fork of Cane Creek**
  - (Lochill Farm MP pages 25-28)
- **UT to Lake Wheeler**
  - (Lyle Creek MP pages 33-34)
  - (Owl's Den MP pages 48-50)
- **Westbrook Lowlands**
  - (Lyle Creek MP pages 33-34)
  - (Owl's Den MP pages 48-50)
- **Long Branch**
  - (Sandy Bridge Farm MP page 196)
  - (Underwood page 37-38) (Candy Creek MP pages 33-35)
  - (Deep Meadow MP pages 149-150)
  - (Martin Dairy MP page 18)
- **UT to Mitchell River (Mickey Creek) and 2 other Martins Creek II RRs**
  - (Mickey Creek- parameters can be found in UT to Martins Creek MP pages 43-47)
- **Toms Creek**
  - (Middle South Muddy Creek MP pages 90-91)
- **UT to Cypress Creek**
  - (Muddy Run II MP page 310)
  - (Muddy Run MP page 124)
- **Bobs Creek**
  - (Neighbors Branch/ Walton Crawley Branch MP pages 16 and 18)
- **Scout West 1, East 2, West 2**
  - (Devil's Racetrack MP pages 45-47)
- **Foust Creek Upstream**

  - (Deep Meadow MP pages 149-150)
  - (Foust Creek MP pages 30-32)
  - (Martin Dairy MP page 18)
- **UT to Underwood Creek**
  - (Newtown MP pages 37-39)
  - (601 East MP pages 90-93)
- **Talbott's Creek**
  - (PeeDee MP pages 202-203)
- **UT to Grassy Creek**
  - (Poplin Ridge MP pages 176-177)
- **UT to West Branch Rocky River**
  - (Roses Creek MP pages 46, 49, 51)
- **Roses Creek Upstream**
  - (Roses Creek MP page 43)
- **Shadrick Creek Reach**
  - (Shadrick Creek MP pages 21, 31, 46, 52, 53, 57)
- **UT to Bailey Creek**
  - (UT to Mill Swamp MP pages 105-106)
- **UT to South Creek**
  - (UT to Mill Swamp MP pages 105-106)
- **UTs to Brice Creek**
  - (UT to Mill Swamp MP pages 105-106)
- **Rockwell Pastures UT4**
  - (Tribs of Wicker Branch- page 113)
- **UT3 to Silver Creek**
  - (upper silver creek page 323 and 324)
- **Basin Creek**
  - (Upper South Hominy MP pages 138-140)
  - (Little Pine Creek II MP pages 12-13)
- **UT to Clarke Creek RRs**
  - (UT to Clarke Creek MP pages 46, 114, 115)
- **UT 1**
  - (UT Clarke Creek MP pages 46, 114, 115)
- **UT to Wildcat Branch**
  - (UT Millers Creek MP page 66)
- **Johnsons Mill Run**
  - (UT Neuse- The Big Ditch MP page 25)
- **UT to Wells Creek**
  - (UT to Cane Creek MP page 156)
- **Brush Creek**
  - (Vile Creek MP pages 58-59)
- **West Fork Chestnut Creek**

    - (Vile Creek MP pages 58-59
- **UT to Little Glade Creek**
    - (Vile Creek MP pages 58-59)
- **Pilot Mountain Tributary**
    - (Lone Hickory MP pages 42 and 44)
- **UT to UT1 of Cedar Creek Project**
    - (Cedar Creek MP page 275)
- **Vile Preserve** (exact location not found)
    - (Lone Hickory MP pages 44-47)
    - (Owl's Den MP pages 48-50)
    - (Henry Fork MP pages 47-48)

- Haw River State Park RR
- Meadow Creek- not yet found on Google Earth
    - (Vile Creek MP pages 58-59- not sufficient parameter information given in table)
- Club Gap Branch- unable to be found on Google Earth
    - (Fletcher MP page 48)
- Spring Creek
- Normans Pasture II RR
- UT 2 (Buffalo Creek RR?)
    - (Buffalo Creek MP pages 53-55)
- UT to Brick Bound Swamp- location not yet determined on Google Earth
    - (UT Millers Creek MP page

Appendix B.

Reference Reach Database

Table X. *Reference Reach database created from North Carolina Division of Mitigation Services (DMS) mitigation plans released from 2009 to June 2018. The associated county, Rosgen stream type, drainage area (mi²), valley slope (ft/ft), and median grain size (D50) (mm) are listed for each site. "U" = data unknown.*

Reference Reach Site Name	County	Stream Type (Rosgen)	Drainage Area (mi^2)	Valley Slope (ft/ft)	D50 (mm)/ substrate
Agony Acres UT1 Reach 3	Guilford	E4	0.3	0.01-0.034	U
Agony Acres UT1A Reach 1	Guilford	E4	0.3	0.01	U
Basin Creek	Wilkes	C4	6.8	0.0139	38.5
Bent Creek	Buncombe	B4	2.35	U	18-33
Bobs Creek	McDowell	C4	0.67	0.025	U
Box Creek	Rutherford	C4	2.13	0.0225	U
Boyd Branch	Buncombe	E4	0.9	0.012	gravel
Brookhaven	Wake	C4	0.14	0.028	16
Brush Creek	Alleghany	C4	1.67	U	U
Buckhorn Creek	Wake	E4	0.2	0.015	25.6
Causey Farm	Guilford	E5	0.63	0.0077	sand-dominated
Cedarock	Alamance	E4b	0.21	0.031	gravel
Cold Springs Creek	Hyde	B4	2.63	0.03	20-46
Collins Creek	Orange	E4	1.68	U	U
Cripple Creek	Alamance	E4	0.17	0.0061	gravel
Dutchman's Creek	Montgomery	B4c	2.9	0.016	32
Foust Creek Upstream	Alamance	C4	1.38	0.0095	gravel bed
Group Camp Tributary	Iredell	E5b	0.1	0.0229	0.3/ sand
Hall Creek	Burke	B4c	4.09	0.0107	13.0/ medium gravel
Henry Fork	Burke	B4a	0.2	0.046	U
Johanna Creek	Johnston	E5/ C5	0.9	0.0027	U
Johnson Mill Run	Pitt	B5	13.5	0.0011	sand-dominated
Little Glade Creek	Alleghany	C4	3.3	U	U
Little Pine Creek III UT2A	Alleghany	A/B4/1	0.12	U	U
Long Branch	Orange	C/E4	1.49	0.006	7.6, 41.6

Meadow Fork	Alleghany	E4	4.37	U	31
Mill Branch	Surry	C4	5	0.0127	20 (bar)-40 (riffle)
Mingo Creek	Wake	C/E5	4	0.003	0.7
Morgan Branch/ Creek	Orange	C4	8.35	U	3.0/ very fine gravel
Pilot Mountain Tributary	Surry	B4	0.27	0.0404	20.1/ coarse gravel
Richland Creek	Moore	C4	1	0.0136	45.0/ very coarse gravel
Rockwell Pastures UT4	Stanly	C4	0.11	0.0173	12.7
Roses Creek Upstream	Burke	C4	4.66	0.008	sand-dominated
Scout East 2	Johnston	E5	0.34	0.005	U
Scout West 1	Johnston	E/C5b	0.06	0.029	U
Scout West 2	Johnston	E5	0.67	0.02	U
Shadrick Creek Upstream	McDowell	E4	2.5	0.016	40
South Fork Mills River	Transylvania	E4	0.72	U	30-42
Spencer Creek (1) - upstream	Montgomery	E4/ C4	0.5	0.0139	8.6/ medium gravel
Spencer Creek (2) - downstream	Montgomery	E4	0.96	0.0109	8.8/ medium gravel
Spencer Creek (3)	Montgomery	E4	0.37	0.022-0.031	11.0/ medium gravel
Talbott's Creek	Randolph	B4c	0.42	0.017	58
Toms Creek	McDowell	C4	3.33	U	31
Upstream UT1 to Henry Fork		B4a	0.05	0.046	34/ cobble (riffle)
UT to Belews Creek	Rockingham	E5	3.4	0.008	U
UT to Bennetts Creek	Gates	C5/6	0.92	0.004	U
UT to Catawba River (1)	Alexander	E5	1.6	0.0058	1.8/ very coarse sand
UT to Catawba River (2)	Catawba	E3b/C3b	1.6	0.0296	75.9/ small cobble
UT to Clarke Creek RR	Mecklenburg	B4c	0.41	U	11.82
UT to Cypress Creek	Duplin	E5	0.47	U	Fine sand
UT to Fisher River	Surry	B4c	0.38	0.016	5.8-7.7/ gravel
UT to Gap Branch	Rutherford	B4a/ A4	0.04	0.1176	19.0/ coarse gravel

UT to Grassy Creek	Union	E4	0.67	U	11
UT to Irish Buffalo Creek	Rowan	E4	0.16	0.009	"cobble/ gravel riffles present"
UT to Kelly Branch	McDowell	A4	0.08	0.0491	9.4/ medium gravel
UT to Lake Wheeler	Wake	E4	0.4	0.01	2.6/ very fine gravel
UT to Lyle Creek (1)	Catawba	C5	0.25	0.0082	0.2/ fine sand
UT to Lyle Creek (2)	Catawba	C5	0.25	0.0045-0.0057	0.2/ very coarse sand
Mickey Reach	Surry	B4	0.45	0.0397	39
UT to Polecat Creek	Randolph	E4	0.41	0.017	16
UT to Rays Fork	Union	E4	0.19	0.016	17.3/ coarse gravel
UT to Reedy Fork	Alamance	C4/1	0.33	0.0075	"0.2 (existing)/ 4.0 (MY4)"
UT to Richardson Creek	Union	B4/ C4b	0.144	0.021	18.6 to 28.9
UT to Richland Creek (1)	Moore	C4/E4	0.28	0.016	U
UT to Richland Creek (2)	Moore	C4/E4	0.97	0.016	46
UT to Rocky Creek	Montgomery	E4b	1.05	0.0261	22.6/ coarse gravel
UT to Sandy Run	Cleveland	E4	0.15	0.02	19.0/ coarse gravel
UT to South Crowders	Gaston	E4	0.22	0.0257	19.7/ coarse gravel
UT to South Fork of Cane Creek	Alamance	C4	0.41	U	11.26/ medium gravel
UT to Spencer Creek	Montgomery	E5	0.014	0.0081	1.0/ coarse sand
UT to Underwood Creek	Union	E4/C4	0.43	0.0065	20.4-38.1
UT to UT1 of Cedar Creek	Sampson	E/C5	0.13	U	medium/coarse sand
UT to Varnals Creek (1)	Alamance	B4/1a	0.24	0.0458	8
UT to Varnals Creek (2)	Alamance	E4	0.41	0.02	U
UT to Wells Creek	Alamance	C4/1	0.13	0.028	4.5
UT to West Branch Rocky River	Iredell	C5	0.07	0.016	gravel-dominated

UT to Wildcat Branch	Lee	E5	0.44	0.0027	sand-dominated
UT to Wilkinson Creek	Chatham	B4c	0.15	0.0017	13.3
UT(s) to Cane Creek	Rowan	C4/E4	0.29	0.0262	27.8/ coarse or medium gravel
UT1 (Clarke Creek Project)	Mecklenburg	B4c	0.38	U	5.02
UT3 to Upper Silver Creek	Burke	E/Bc	0.12	0.0188	U
Vile Preserve	Catawba	E5	1.09	0.0074	0.4/ medium sand
West Fork Chestnut Creek	U (Virginia)	E4	1.6	U	U
Westbrook Lowlands	Johnston	E/C5	0.9	0.0027	0.7/ coarse sand
Basin Creek	Wilkes	C4	7.2	0.01437	58
Cabin Branch UT	Durham	C4	1.256	0.0118	U
East Prong Hickey Fork	Madison	B3a	1.98	0.045	75
Goshen Branch	Watauga	B3a	1.45	U	69
Hogans Creek UT	Caswell	E5	0.5	0.00693	0.2
Raccoon Creek	Haywood	E5	2.859	0.0109	0.75
Riverbend Park	Catawba	U	0.54	U	U
UT Gap Branch	Rutherford	B4a/ A4	0.04	0.068	19
UT Lake Norman (Group Camp)	Iredell	E5b	0.1	0.0167	0.3
UT to Beaverdam Creek	Davidson	U	0.34	U	U
UT to Catawba River	Catawba	E5-E3b/C3b	1.6	0.0051-0.0287	1.8-75.9
UT to Look Shoals (smaller)	Iredell	U	0.27	U	U
UT to Lookout Shoals	Iredell	U	0.8	U	U
UT to Lyle Creek	Catawba	C5	0.25	0.005	0.2
UT to South Crowders	Gaston	E4	0.22	0.0091	19.7
Vile Preserve	Catawba	E5	1.09	0.0068	0.4
Worley Creek	Watauga	E5	0.4	0.0075	0.27
UT Grassy Creek (Poplin Ridge)	Union	E4	0.66	0.005	11
UT to Hauser Creek (John Sparks Property)	Yadkin	U	0.07	U	U

UT to Landrum (@Zeb Ferguson Rd)	Chatham	U	1	U	U
UT to Landrum Creek (@Elmer Keck Rd)	Chatham	C4	0.253	0.0077	U
UT to McLendons Crk (Stonebridge)	Moore	E5	0.48-0.77	0.0025	0.75
UT to Smitheys Creek	Wilkes	U	0.18	U	U
UT to Lake Jeannette	Guilford	C4	0.25	0.0076	U
UT to Varnals Creek	Alamance	C4/1	0.4	0.025	8.7
UT to Polecat Creek	Randolph	E4	0.4	0.017	7.1
Fork Creek upstream	Randolph	B4c	2.2	U	33, 28
Fork Creek downstream	Randolph	B4c	U	U	U
UT to Cane Creek A3	Alamance	E4	0.88	0.013	8.9
Landrum Creek	Chatham	C4	2.53	0.008	U
Morgan Creek	Orange	C4	U	0.007	3
UT to SW Prong Beaver Dam	Wake	C5	0.28	0.03	1
Big Branch	Surry	E4	1.9	0.0087	U
Coffey Creek	Meckenburg	B4c	4.04	0.0109	2.3
Glade Creek	Alexander	B4	1.7	0.012	U
Johnsons Mill Run	Pitt	B5c	13.5	0.001	U
Little Beaver Creek	Wake	C5	0.3	U	U
Little Rockfish Creek	Cumberland	E5	16.5	0.0012	U
Long Creek	Meckenburg	C3 to C5	10.9	0.0045	84(riffle)
Lost Cove Creek	Avery	C3	24.8	0.0084	144
Mitchell River	Surry	B4c	6	0.009	U
Moseley Creek	Lenoir	E	8	0.0015	U
North Prong Creek	Durham	C5	3.15	0.0023	0.2
Richland Creek	Moore	C4	1	0.0136	45
UT to Back Creek	Randolph	B4c	0.38	U	U
UT to Cabin Branch	Durham	C4/E4	U	U	11.4
UT to Park South Drive	Meckenburg	E5	0.16	0.02	0.82

Appendix C.

Pre-Coding Design Interface Plans (Hand-drawn)

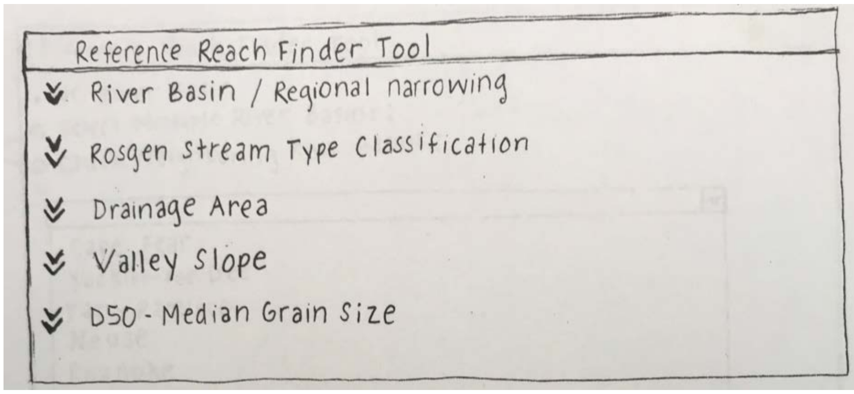

Figure X. *Original collapsed categorized script tool design interface for the Reference Reach Finder Tool.*

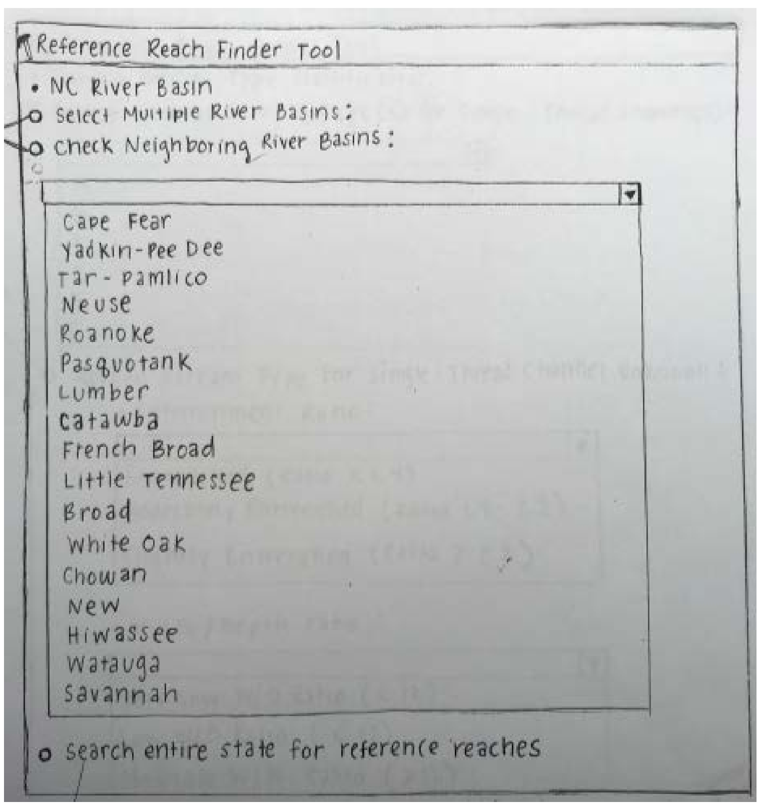

Figure X. *Original river basin selection design interface for the Reference Reach Finder Tool.*

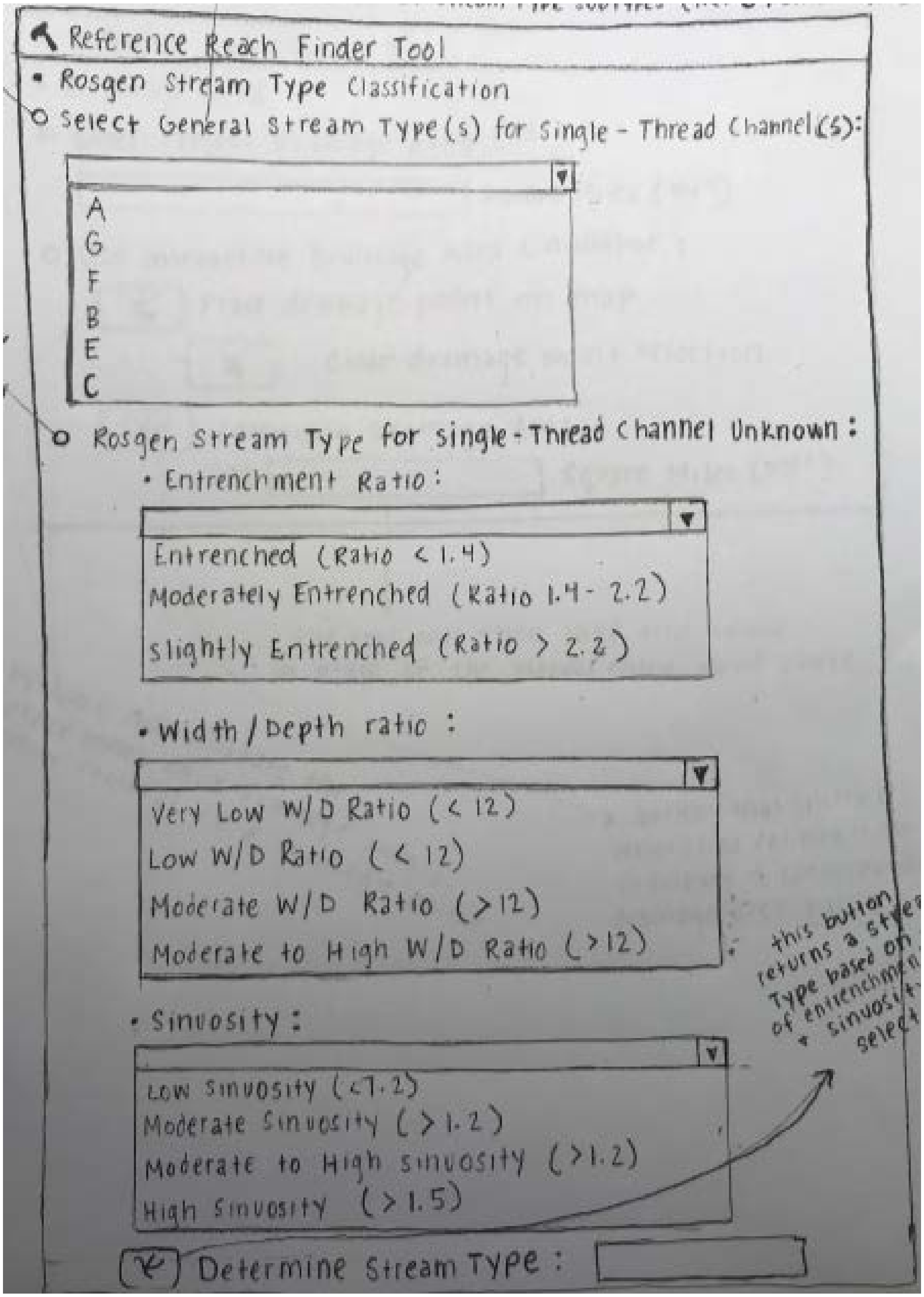

Figure X. *Original Rosgen Stream Type selection design interface for the Reference Reach Finder Tool.*

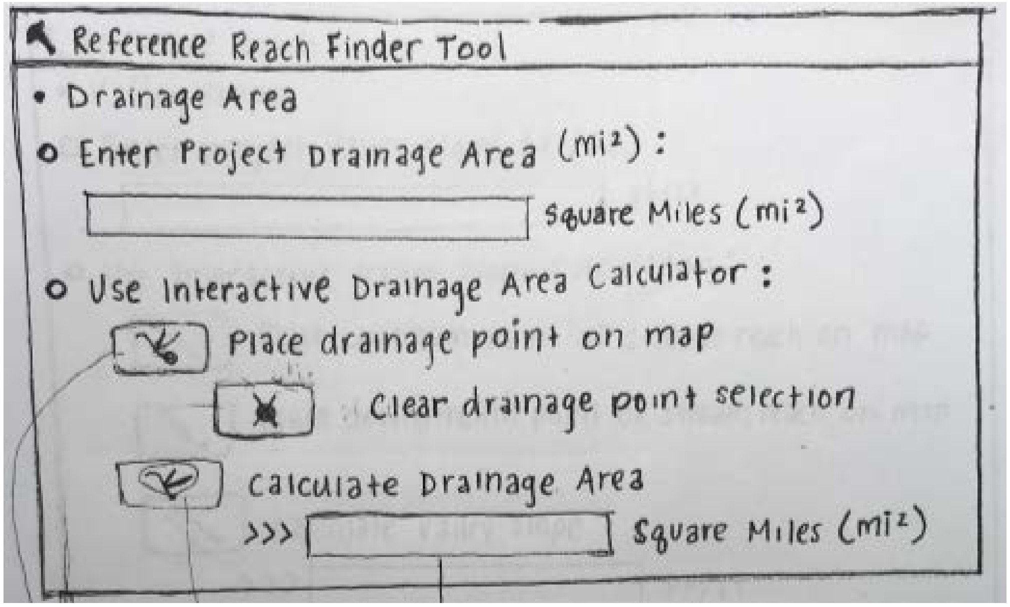

Figure X. *Original drainage area entry design interface for the Reference Reach Finder Tool.*

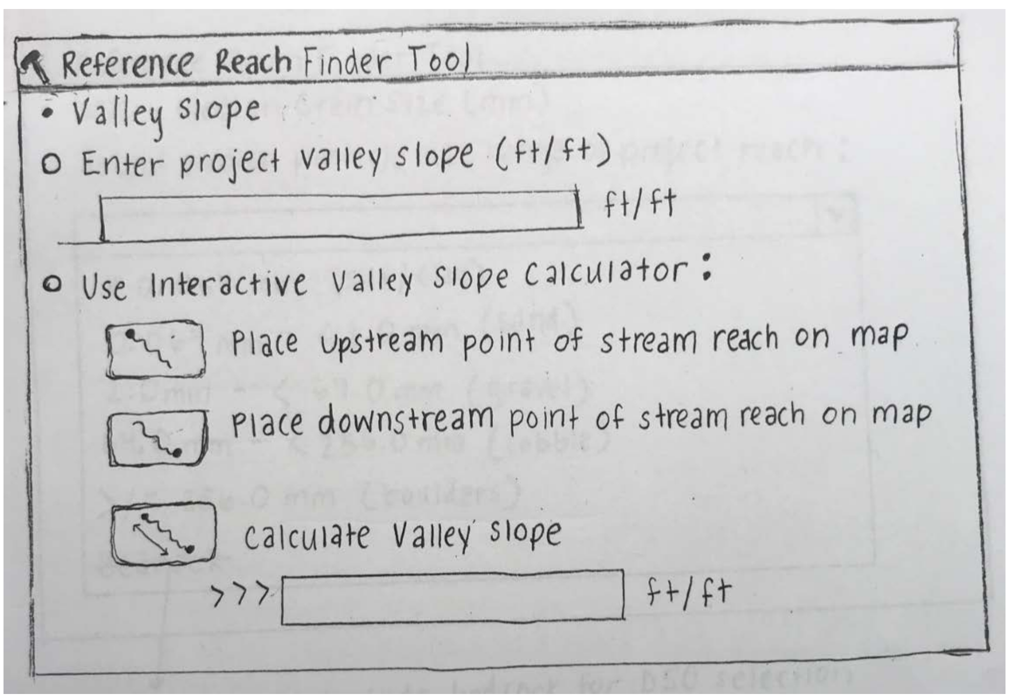

Figure X. *Original valley slope entry design interface for the Reference Reach Finder Tool.*

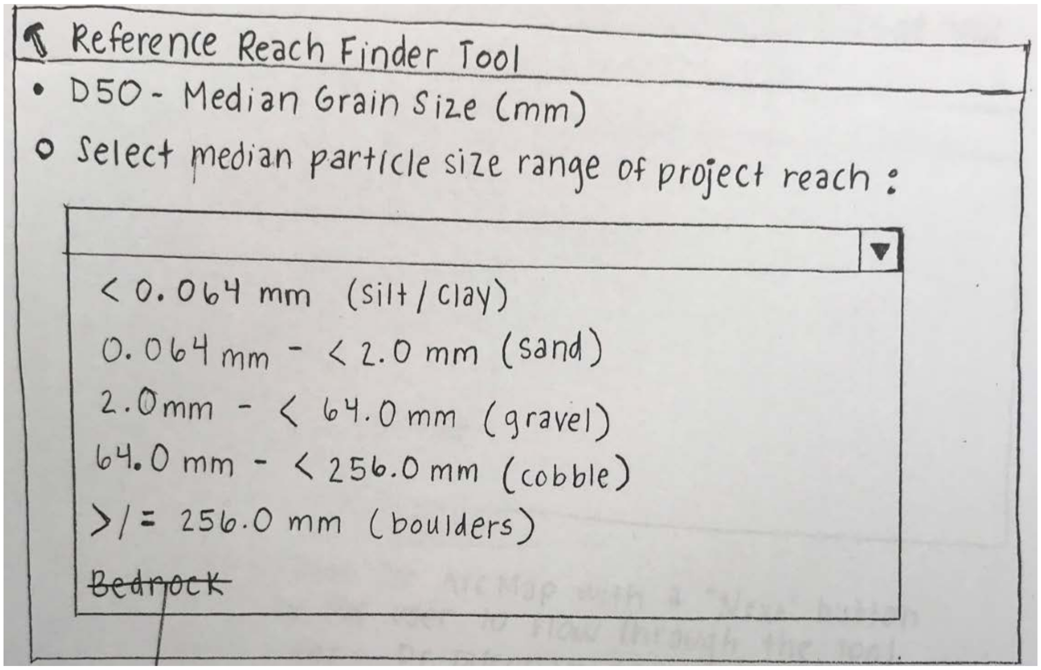

Figure X. *Original Median Grain Size (D50) selection design interface for the Reference Reach Finder Tool*

Appendix D

Final Python Script used for Python Toolbox

Created in Python Win IDE

```
# Clare Maher
# Master of Natural Resources Final Project: Reference Reach Finder Tool
# June 2019

import arcpy, os

def getAbsPath(relativePath):
    '''''Return the absolute path given a relative path to this file'''
    tbxPath = os.path.abspath(__file__)
    tbxDir = os.path.dirname(tbxPath)
    fullPath = os.path.join(tbxDir, relativePath)
    return os.path.abspath(fullPath)

def printArc(message):
    '''''Print message for script tool and standard output.'''
    print message
    arcpy.AddMessage(message)


class Toolbox(object):
    def __init__(self):
        """Define the toolbox (the name of the toolbox is the name of the
        .pyt file)."""
        self.label = "ReferenceReachFinderTool"
        self.alias = "ReferenceReachFinderTool"

        # List of tool classes associated with this toolbox
        self.tools = [Tool]


class Tool(object):
    def __init__(self):
        """Define the tool (tool name is the name of the class)."""
        self.label = "Reference Reach Finder Tool"
        self.description = '''''A reference reach is a stable stream channel
        segment that serves as a model for restoration, and is described by
        quantitative field data of its morphological features (Rosgen 1998).
        The Reference Reach Finder (RRF) tool gathers information about the
        river basin, stream type, drainage area, valley slope and median grain
        size in order to find the most suitable reference reach for any project
        in North Carolina. Additionally, the RRF tool provides user flexibility
        by allowing input data from other states as well. The RRF tool is intended
        to be used by restoration designers with experience in natural channel design;
        however, several features are included to aid the user when input values are unknown.'''

        self.canRunInBackground = False


    def getParameterInfo(self):
        """Define parameters"""

        # Select a workspace
        param0 = arcpy.Parameter()
        param0.name = 'Select_workspace_containing_reference_reach_data'
```

```
        param0.displayName = 'Select Workspace Containing Reference Reach Data'
        param0.parameterType = 'Required'
        param0.direction = 'Input'
        param0.datatype = 'DEWorkspace'

        # Select an output directory
        param1 = arcpy.Parameter()
        param1.name = 'Select_Output_Directory'
        param1.displayName = 'Select Output Directory'
        param1.parameterType = 'Required'
        param1.direction = 'Input'
        param1.datatype = 'DEWorkspace'
        relativePath0 = '../data/output/'
        param1.value = getAbsPath(relativePath0)

        # Select reference reach point shapefile
        param2 = arcpy.Parameter()
        param2.name = 'Reference_Reach_Points_File'
        param2.displayName = 'Select Reference Reach Points File'
        param2.parameterType = 'Required'
        param2.direction = 'Input'
        param2.datatype = 'DEShapefile'
        relativePath1 = '../data/NC_Reference_Reaches/NC_Reference_Reach_Points.shp'
        param2.value = getAbsPath(relativePath1)
        param2.filter.list = ['.shp']

        # Select river basins shapefile
        param3 = arcpy.Parameter()
        param3.name = 'Select_River_Basins_File'
        param3.displayName = 'Select River Basins File'
        param3.parameterType = 'Optional'
        param3.direction = 'Input'
        param3.datatype = 'DEShapefile'
        relativePath2 = '../data/River_Basins/NC/NC_RB_Project.shp'
        param3.value = getAbsPath(relativePath2)
        param3.filter.list = ['.shp']

        # Select field in river basins file containing river basin names
        param4 = arcpy.Parameter()
        param4.name = 'Field_Containing_River_Basin_Names'
        param4.displayName = 'Field Containing River Basin Names'
        param4.parameterType = 'Optional'
        param4.direction = 'Input'
        param4.datatype = 'Field'
        param4.defaultEnvironmentName = 'Name'
        param4.value = 'Name'
        ##param4.filter.list = ['Text']
        param4.parameterDependencies = [param3.name]

        # Select river basins of interest
        param5 = arcpy.Parameter()
        param5.name = 'Select_River_Basins_Of_Interest'
        param5.displayName = 'Select River Basin(s) of Interest'
        param5.parameterType = 'Optional'
        param5.direction = 'Input'
        param5.datatype = 'GPString'
        param5.multiValue = True
        param5.filter.type = 'ValueList'
        param5.filter.list = []
        ##param5.parameterDependencies = [param4.name]

```

```
        # Output river basin selection feature class
        param6 = arcpy.Parameter()
        param6.name = 'Output_River_Basin_Selection_File'
        param6.displayName = 'Output River Basin Selection File'
        param6.parameterType = 'Derived'
        param6.direction = 'Output'
        param6.datatype = 'DEFeatureClass'
        param6.value = 'River_Basin_Selection'
        param6.defaultEnvironmentName = '../data/output/River_Basin_Selection'

        # Output reference reach point feature class from selected river basin(s)
        param7 = arcpy.Parameter()
        param7.name = 'Output_Reference_Reach_Points_River_Basin_Selection'
        param7.displayName = 'Output Reference Reach Points from River Basin Selection'
        param7.parameterType = 'Derived'
        param7.direction = 'Output'
        param7.datatype = 'DEFeatureClass'
        param7.value = 'RR_Points_In_River_Basin'
        param7.defaultEnvironmentName = '../data/output/RR_Points_In_River_Basin'

        # Select stream type from list of Rosgen stream types for for single thread channels (if k
nown)
        param8 = arcpy.Parameter()
        param8.name = 'Select_Rosgen_Stream_Type_For_Single_Thread_Channels'
        param8.displayName = 'Select Rosgen Stream Type(s) for Single Thread Channels'
        param8.parameterType = 'Optional'
        param8.direction = 'Input'
        param8.datatype = 'String'
        param8.multiValue = True
        param8.filter.list = ["A", "G", "F", "B", "E", "C"]
        param8.defaultEnvironmentName = 'C'

        # Select field in reference reach shapefile containing stream type
        param9 = arcpy.Parameter()
        param9.name = 'Field_Containing_Stream_Type'
        param9.displayName = 'Field in Reference Reach Points File Containing Stream Type'
        param9.parameterType = 'Optional'
        param9.direction = 'Input'
        param9.datatype = 'Field'
        param9.defaultEnvironmentName = 'ST_Letter'
        param9.value = 'ST_Letter'
        param9.parameterDependencies = [param2.name]

        # Unknown stream type: Select entrenchment ratio
        param10 = arcpy.Parameter()
        param10.name = 'Entrenchment_Ratio'
        param10.displayName = 'Stream Type Unknown: Select Entrenchment Ratio'
        param10.parameterType = 'Optional'
        param10.direction = 'Input'
        param10.datatype = 'String'
        param10.filter.list = ["Slightly Entrenched (Ratio> 2.2)", "Moderately Entrenched (Ratio 1
.4 - 2.2)", "Entrenched (Ratio > 1.4)"]
        param10.defaultEnvironmentName = "Slightly Entrenched (Ratio> 2.2)"

        # Unknown stream type: Select width to depth ratio
        param11 = arcpy.Parameter()
        param11.name = 'Width_To_Depth_Ratio'
        param11.displayName = 'Stream Type Unknown: Select Width To Depth Ratio'
        param11.parameterType = 'Optional'
        param11.direction = 'Input'
        param11.datatype = 'String'
```

```
        param11.filter.list = ["Very Low Width to Depth Ratio (< 12)", "Low Width to Depth Ratio (< 12)", "Moderate Width to Depth Ratio (> 12)", "Moderate to High Width to Depth Ratio (> 12)"]
        param11.defaultEnvironmentName = "Very Low Width to Depth Ratio (< 12)"

        # Unknown stream type: Select sinuosity
        param12 = arcpy.Parameter()
        param12.name = 'Sinuosity'
        param12.displayName = 'Stream Type Unknown: Select Sinuosity'
        param12.parameterType = 'Optional'
        param12.direction = 'Input'
        param12.datatype = 'String'
        param12.filter.list = ["Low Sinuosity (< 1.2)", "Moderate Sinuosity (>  1.2)", "Moderate to High Sinuosity (>  1.2)", "High Sinuosity (> 1.5)"]
        param12.defaultEnvironmentName = "Low Sinuosity (< 1.2)"

        # Output reference reach point feature class from stream type selection (output)
        param13 = arcpy.Parameter()
        param13.name = 'Output_Reference_Reach_Points_Stream_Type_Selection'
        param13.displayName = 'Output Reference Reach Points Stream Type Selection'
        param13.parameterType = 'Derived'
        param13.direction = 'Output'
        param13.datatype = 'DEFeatureClass'
        param13.value = 'Stream_Type_Selection'
        param13.defaultEnvironmentName = '../data/output/Stream_Type_Selection'

        # Enter drainage area if known
        param14 = arcpy.Parameter()
        param14.name = 'Enter_Drainage_Area_(Square_Miles)'
        param14.displayName = 'Enter Drainage Area (Square Miles)'
        param14.parameterType = 'Optional'
        param14.direction = 'Input'
        param14.datatype = 'Double'
        param14.defaultEnvironmentName = '0.0'

        # Select field in reference reach shapefile containing drainage area
        param15 = arcpy.Parameter()
        param15.name = 'Reference_Reach_Field_Containing_Drainage_Area'
        param15.displayName = 'Field in Reference Reach Points File Containing Drainage Area'
        param15.parameterType = 'Optional'
        param15.direction = 'Input'
        param15.datatype = 'Field'
        param15.defaultEnvironmentName = 'Drainage_A'
        param15.value = 'Drainage_A'
        ##param15.filter.list = ['Double', 'Long']
        param15.parameterDependencies = [param2.name]

        # Drainage area range derived from user input drainage area value
        param16 = arcpy.Parameter()
        param16.name = 'Drainage_Area_Range_Calculated_For_Reference_Reach_Selection'
        param16.displayName = 'Drainage Area Range Calculated for Reference Reach Selection'
        param16.parameterType = 'Derived'
        param16.direction = 'Output'
        param16.datatype = 'Double'
        param16.defaultEnvironmentName = '0.0'
        # Did I write parameters correctly for this type of derived output?

        # Output reference reach point feature class from drainage area selection
        param17 = arcpy.Parameter()
        param17.name = 'Output_Reference_Reach_Points_Drainage_Area_Selection'
        param17.displayName = 'Output Reference Reach Points from Drainage Area Selection'
        param17.parameterType = 'Derived'
```

```python
            param17.direction = 'Output'
            param17.datatype = 'DEFeatureClass'
            param17.value = 'Drainage_Area_Selection'
            param17.defaultEnvironmentName = '../data/output/Drainage_Area_Selection'

            # Interactive drainage area calculator feature: User input DEM file (raster)
            param18 = arcpy.Parameter()
            param18.name = 'Select_DEM_Raster_File_In_Workspace'
            param18.displayName = 'Select DEM Raster File In Workspace'
            param18.parameterType = 'Optional'
            param18.direction = 'Input'
            param18.datatype = 'DERasterDataset'
            relativePath3 = '../data/DEMs/ralRastTest.tif'
            param18.value = getAbsPath(relativePath3)
            ##param18.filter.list = ['.tif']

            # Interactive drainage area calculator feature: User input drainage point file
            param19 = arcpy.Parameter()
            param19.name = 'Select_Drainage_Point_File_In_Workspace'
            param19.displayName = 'Select Drainage Point File In Workspace'
            param19.parameterType = 'Optional'
            param19.direction = 'Input'
            param19.datatype = 'DEShapefile'
            relativePath4 = '../data/Drainage_Points/Rocky_Branch_Drainage_Point.shp'
            param19.value = getAbsPath(relativePath4)
            param19.filter.list = ['.shp']

            # Fill tool output (raster)
            param20 = arcpy.Parameter()
            param20.name = 'Fill_Tool_Output_Raster_File'
            param20.displayName = 'Fill Tool Output Raster File'
            param20.parameterType = 'Derived'
            param20.direction = 'Output'
            param20.datatype = 'DERasterDataset'
            param20.value = '/data/output/fill.tif'
            param20.defaultEnvironmentName = '../data/output/fill.tif'

            # Flow Direction tool output (raster)
            param21 = arcpy.Parameter()
            param21.name = 'Flow_Direction_Tool_Output_Raster_File'
            param21.displayName = 'Flow Direction Tool Output Raster File'
            param21.parameterType = 'Derived'
            param21.direction = 'Output'
            param21.datatype = 'DERasterDataset'
            param21.value = '/data/output/flowDir.tif'
            param21.defaultEnvironmentName = '../data/output/flowDir.tif'

            # Flow Accumulation tool output (raster)
            param22 = arcpy.Parameter()
            param22.name = 'Flow_Accumulation_Tool_Output_Raster_File'
            param22.displayName = 'Flow Accumulation Tool Output Raster File'
            param22.parameterType = 'Derived'
            param22.direction = 'Output'
            param22.datatype = 'DERasterDataset'
            param22.value = '/data/output/flowAcc.tif'
            param22.defaultEnvironmentName = '../data/output/flowAcc.tif'

            # Flow Accumulation raster calculation output (raster)
            param23 = arcpy.Parameter()
            param23.name = 'Flow_Accumulation_Raster_Calculation_Output_File'
            param23.displayName = 'Flow Accumulation Raster Calculation Output File'
```

```
        param23.parameterType = 'Derived'
        param23.direction = 'Output'
        param23.datatype = 'GPRasterCalculatorExpression'
        param23.value = '/data/output/calculateRaster.tif'
        param23.defaultEnvironmentName = '../data/output/calculateRaster.tif'

        # Snap Pour Point tool output (raster)
        param24 = arcpy.Parameter()
        param24.name = 'Snap_Pour_Point_Tool_Output_Raster_File'
        param24.displayName = 'Snap Pour Point Tool Output Raster File'
        param24.parameterType = 'Derived'
        param24.direction = 'Output'
        param24.datatype = 'DERasterDataset'
        param24.value = '/data/output/snapDrainPt.tif'
        param24.defaultEnvironmentName = '../data/output/snapDrainPt.tif'

        # Watershed tool output (raster)
        param25 = arcpy.Parameter()
        param25.name = 'Watershed_Tool_Output_Raster_File'
        param25.displayName = 'Watershed Tool Output Raster File'
        param25.parameterType = 'Derived'
        param25.direction = 'Output'
        param25.datatype = 'DERasterDataset'
        param25.value = '/data/output/watershed.tif'
        param25.defaultEnvironmentName = '../data/output/watershed.tif'

        # Raster to Polygon tool output (shapefile)
        param26 = arcpy.Parameter()
        param26.name = 'Delineated_Watershed_Raster_To_Polygon_Tool_Output_File'
        param26.displayName = 'Delineated Watershed: Raster To Polygon Output File'
        param26.parameterType = 'Derived'
        param26.direction = 'Output'
        param26.datatype = 'DEShapefile'
        param26.value = '/data/output/outWSPolygon.shp'
        param26.defaultEnvironmentName = '../data/output/outWSPolygon.shp'

        # Drainage area range derived from interactive drainage area calculator feature
        param27 = arcpy.Parameter()
        param27.name = 'Interactive_Drainage_Area_Calculator_Selection_Range'
        param27.displayName = 'Interactive Drainage Area Calculator Selection Range'
        param27.parameterType = 'Derived'
        param27.direction = 'Output'
        param27.datatype = 'Double' # Is this the correct data type?
        param27.defaultEnvironmentName = '../data/output/DrainageAreaRangeCalc'

        # Output reference reach point feature class from interactive drainage area calculator sel
ection
        param28 = arcpy.Parameter()
        param28.name = 'Output_Reference_Reach_Points_Interactive_Drainage_Area_Selection'
        param28.displayName = 'Output Reference Reach Points from Interactive Drainage Area Select
ion'
        param28.parameterType = 'Derived'
        param28.direction = 'Output'
        param28.datatype = 'DEFeatureClass'
        param28.value = 'Interactive_Drainage_Area_Output'
        param28.defaultEnvironmentName = '../data/output/Interactive_Drainage_Area_Selection'

        # Enter valley slope if known
        param29 = arcpy.Parameter()
        param29.name = 'Enter_Valley_Slope_(ft/ft)'
        param29.displayName = 'Enter Valley Slope (ft/ft)'
```

```
        param29.parameterType = 'Optional'
        param29.direction = 'Input'
        param29.datatype = 'Double'
        param29.defaultEnvironmentName = '0.0'

        # Select field in reference reach shapefile containing valley slope
        param30 = arcpy.Parameter()
        param30.name = 'Reference_Reach_Field_Containing_Valley_Slope'
        param30.displayName = 'Field In Reference Reach Points File Containing Valley Slope'
        param30.parameterType = 'Optional'
        param30.direction = 'Input'
        param30.datatype = 'Field'
        param30.defaultEnvironmentName = 'V_Slope'
        param30.value = 'V_Slope'
        ##param30.filter.list = ['Text']
        param30.parameterDependencies = [param2.name]

        # Valley slope selection range derived from user input valley slope value
        param31 = arcpy.Parameter()
        param31.name = 'Valley_Slope_Selection_Range_from_Input_Valley_Slope_Value'
        param31.displayName = 'Valley Slope Selection Range from Input Valley Slope Value'
        param31.parameterType = 'Derived'
        param31.direction = 'Output'
        param31.datatype = 'Double' #Correct data type for derived range values?
        param31.defaultEnvironmentName = '../data/output/ValleySlopeSelectionRange'

        # Output reference reach point feature class from valley slope selection
        param32 = arcpy.Parameter()
        param32.name = 'Output_Reference_Reach_Points_Valley_Slope_Selection'
        param32.displayName = 'Output Reference Reach Points from Valley Slope Selection'
        param32.parameterType = 'Derived'
        param32.direction = 'Output'
        param32.datatype = 'DEFeatureClass'
        param32.value = 'Valley_Slope_Selection'
        param32.defaultEnvironmentName = '../data/output/Valley_Slope_Selection'

        # Interactive valley slope calculator feature: contour lines shapefile
        param33 = arcpy.Parameter()
        param33.name = 'Input_Contour_Lines_File'
        param33.displayName = 'Input Contour Lines File'
        param33.parameterType = 'Optional'
        param33.direction = 'Input'
        param33.datatype = 'DEShapefile'
        relativePath5 = '../data/Contours/NC/Wake_County/Contour_Wake_Clip.shp'
        param33.value = getAbsPath(relativePath5)

        # Interactive valley slope calculator feature: field in contour lines file containing elev
ation values
        param34 = arcpy.Parameter()
        param34.name = 'Contour_Lines_Field_Containing_Elevation_Values'
        param34.displayName = 'Field In Contour Lines File Containing Elevation Values'
        param34.parameterType = 'Optional'
        param34.direction = 'Input'
        param34.datatype = 'Field'
        param34.value = 'Z_Feet'
        param34.defaultEnvironmentName = 'Z_Feet'
        param34.parameterDependencies = [param33.name]

        # Valley elevation (feet) upstream (due to time constraints, user will need to look at ele
vation labels on contour lines to determine this value.)
        param35 = arcpy.Parameter()
```

```
        param35.name = 'Valley_Elevation_Upstream_(Feet)'
        param35.displayName = 'Valley Elevation Upstream (Feet)'
        param35.parameterType = 'Optional'
        param35.direction = 'Input'
        param35.datatype = 'Long'
        param35.defaultEnvironmentName = '0' #should I set this default value without quotations?

        # Valley elevation (feet) downstream (due to time constraints, user will need to look at e
levation labels on contour lines to determine this value.)
        param36 = arcpy.Parameter()
        param36.name = 'Valley_Elevation_Downstream_(Feet)'
        param36.displayName = 'Valley Elevation Downstream (Feet)'
        param36.parameterType = 'Optional'
        param36.direction = 'Input'
        param36.datatype = 'Long'
        param36.defaultEnvironmentName = '0'

        # Valley length (feet) between valley elevation upstream and valley elevation downstream (
due to time constraints, user will need to use the measure tool and approximate this value).
        param37 = arcpy.Parameter()
        param37.name = 'Valley_Length_Between_Valley_Elevations_(Feet)'
        param37.displayName = 'Valley Length (Feet)'
        param37.parameterType = 'Optional'
        param37.direction = 'Input'
        param37.datatype = 'Double'
        param37.defaultEnvironmentName = '0'

        # Valley slope calculation value derived from valley elevations and valley length (ft/ft)

        param38 = arcpy.Parameter()
        param38.name = 'Valley_Slope_Calculation_(ft/ft)'
        param38.displayName = 'Valley Slope Calculation (ft/ft)'
        param38.parameterType = 'Derived'
        param38.direction = 'Output'
        param38.datatype = 'Double'
        param38.defaultEnvironmentName = '../data/output/vSlopeCalc'

        # Valley slope selection range derived from valley slope calculation value
        param39 = arcpy.Parameter()
        param39.name = 'Valley_Slope_Selection_Range_from_Calculation'
        param39.displayName = 'Valley Slope Selection Range from Calculation'
        param39.parameterType = 'Derived'
        param39.direction = 'Output'
        param39.datatype = 'Double' # Is this the correct data type?

        # Output reference reach point feature class from interactive valley slope calculator sele
ction
        param40 = arcpy.Parameter()
        param40.name = 'Output_Reference_Reach_Points_Valley_Slope_Calculation_Selection'
        param40.displayName = 'Output Reference Reach Points Valley Slope Calculation Selection'
        param40.parameterType = 'Derived'
        param40.direction = 'Output'
        param40.datatype = 'DEFeatureClass'
        param40.value = 'Interactive_Valley_Slope_Selection'
        param40.defaultEnvironmentName = '../data/output/Interactive_Valley_Slope_Selection'

        # Select median grain size (D50) class from list
        param41 = arcpy.Parameter()
        param41.name = 'Select_Median_Grain_Size_(D50)_Class'
        param41.displayName = 'Select Median Grain Size (D50) Class'
```

```
        param41.parameterType = 'Optional'
        param41.direction = 'Input'
        param41.datatype = 'String'
        param41.filter.list = ["< 0.064 mm (Silt/Clay)", "0.064 mm - < 2.0 mm (sand)", "2.0 mm - < 64.0 mm (gravel)", "64.0 mm - < 256.0 mm (cobble)", ">= 256.0 mm (boulders)"]
        param41.defaultEnvironmentName = "2.0 mm - < 64.0 mm (gravel)"

        # Select field in reference reach shapefile containing median grain size classes
        param42 = arcpy.Parameter()
        param42.name = 'Reference_Reach_Points_Field_Containing_Median_Grain_Size_(D50)_Classes'
        param42.displayName = 'Field in Reference Reach Points File Containing Median Grain Size (D50) Classes'
        param42.parameterType = 'Optional'
        param42.direction = 'Input'
        param42.datatype = 'Field'
        param42.defaultEnvironmentName = 'D50_Class'
        param42.value = 'D50_Class'
        ##param42.filter.list = ['Text']
        param42.parameterDependencies = [param2.name]

        # Output reference reach point feature class from D50 selection
        param43 = arcpy.Parameter()
        param43.name = 'Output_Reference_Reach_Points_D50_Selection'
        param43.displayName = 'Output Reference Reach Points D50 Selection'
        param43.parameterType = 'Derived'
        param43.direction = 'Output'
        param43.datatype = 'DEFeatureClass'
        param43.value = 'D50_Selection'
        param43.defaultEnvironmentName = '../data/output/D50_Selection'

# As mentioned in execute section, I have been unsucessful in getting the Merge and final output file to work properly through the Python Toolbox. Worked well through Python Win.
        # Merge tool output file (shapefile)
        param44 = arcpy.Parameter()
        param44.name = 'Merge_of_all_Reference_Reach_Selection_Feature_Classes'
        param44.displayName = 'Merge of all Reference Reach Selection Feature Classes'
        param44.parameterType = 'Derived'
        param44.direction = 'Output'
        param44.datatype = 'DEShapefile'
        param44.value = 'Reference_Reach_Final_Output.shp'
        param44.defaultEnvironmentName = '/data/output/Reference_Reach_Final_Output.shp'

##        # Output reference reach point feature class from merging (and deleting identical) of all previous selections
##        param45 = arcpy.Parameter()
##        param45.name = 'Reference_Reach_Final_Output_Layer'
##        param45.displayName = 'Reference Reach Final Output Layer'
##        param45.parameterType = 'Required'
##        param45.direction = 'Output'
##        param45.datatype = 'DEFeatureClass'
##        param45.value = 'RR_Final_Output'
##        param45.defaultEnvironmentName = '/data/output/RR_Final_Output'


        paramList = [param0, param1, param2, param3, param4, param5, param6, param7, param8, param9, param10, \
                     param11, param12, param13, param14, param15, param16, param17, param18, param19, \
                     param20, param21, param22, param23, param24, param25, param26, param27, param28, \
```

```
                     param29, param30, param31, param32, param33, param34, param35, param36, param
37, \
                     param38, param39, param40, param41, param42, param43, param44]#, param45]

        numParams = len(paramList)

        for index in range(numParams):
            if index in range(3, 7):
                paramList[index].category = '1. River Basin Selection'
            elif index in range(7, 14):
                paramList[index].category = '2. Stream Type Selection/ Unknown Stream Type Determi
nation'
            elif index in range(14, 18):
                paramList[index].category = '3. Drainage Area Entry'
            elif index in range(18, 29):
                paramList[index].category = '4. Interactive Drainage Area Calculator'
            elif index in range(29, 33):
                paramList[index].category = '5. Valley Slope Entry'
            elif index in range(33, 41):
                paramList[index].category = '6. Interactive Valley Slope Calculator'
            elif index in range(41, 44):
                paramList[index].category = '7. Median Grain Size (D50) Selection'

##        for param in paramList:
##            return param
        return paramList

    def isLicensed(self):
        """Allow the tool to run only if the Spatial Analyst extension is enabled """

        """ Spatial Analyst extension only needed if optional 'interactive drainage area
        calculator' feature is used, so licensing check may not be done to allow for tool
        flexibility (i.e. if user wants to use other features of the tool, but does not have
        the Spatial Analyst license.) """

        if arcpy.CheckExtension('Spatial') == 'Available':
            return True
        else:
            return False

    def updateParameters(self, parameters):
        """Modify the values and properties of parameters before internal
        validation is performed.  This method is called whenever a parameter
        has been changed."""

        if parameters[4].value:
            with arcpy.da.SearchCursor(parameters[3].valueAsText, parameters[4].valueAsText) as ro
ws:
                parameters[5].filter.list = sorted(list(set([row[0] for row in rows])))
        else:
            parameters[5].filter.list = []

        if parameters[10].altered:
            if parameters[10].value == "Slightly Entrenched (Ratio> 2.2)":
                parameters[11].filter.list = ["Very Low Width to Depth Ratio (< 12)", "Moderate to
 High Width to Depth Ratio (> 12)"]
                parameters[12].filter.list = ["Moderate to High Sinuosity (>  1.2)", "High Sinuosi
ty (> 1.5)"]
                if parameters[11].altered:
                    if parameters[11].value == "Very Low Width to Depth Ratio (< 12)":
                        parameters[12].filter.list = ["High Sinuosity (> 1.5)"]
```

```
                        if parameters[12].altered:
                            parameters[8].filter.list = ['E']
                            arcpy.AddMessage("Stream Type is E")


                    elif parameters[11].value == "Moderate to High Width to Depth Ratio (> 12)":
                        parameters[12].filter.list = ["Moderate to High Sinuosity (>  1.2)"]
                        if parameters[12].altered:
                            parameters[8].filter.list = ['C']
                            arcpy.AddMessage("Stream Type is C")

            elif parameters[10].value == "Moderately Entrenched (Ratio 1.4 - 2.2)":
                parameters[11].filter.list = ["Moderate Width to Depth Ratio (> 12)"]
                parameters[12].filter.list = ["Moderate Sinuosity (>  1.2)"]
                if parameters[11].altered and parameters[2].altered:
                        parameters[8].filter.list = ['B']
                        arcpy.AddMessage("Stream Type is B")

            elif parameters[10].value == "Entrenched (Ratio > 1.4)":
                parameters[11].filter.list = ["Low Width to Depth Ratio (< 12)", "Moderate to High Width to Depth Ratio (> 12)"]
                parameters[12].filter.list = ["Low Sinuosity (< 1.2)", "Moderate Sinuosity (>  1.2)"]

                if parameters[11].altered:
                    if parameters[11].value == "Moderate to High Width to Depth Ratio (> 12)":
                        parameters[12].filter.list = ["Moderate Sinuosity (>  1.2)"]
                        if parameters[12].altered:
                            parameters[8].filter.list = ['F']
                            arcpy.AddMessage("Stream Type is F")

                    elif parameters[11].value == "Low Width to Depth Ratio (< 12)":
                        if parameters[12].altered:
                            if parameters[12].value == "Low Sinuosity (< 1.2)":
                                arcpy.addMessage("Stream Type is A")
                            if parameters[12].value == "Moderate Sinuosity (>  1.2)":
                                parameters[8].filter.list = ['G']
                                arcpy.AddMessage("Stream Type is G")



        return

    def updateMessages(self, parameters):
        """Modify the messages created by internal validation for each tool
        parameter.  This method is called after internal validation."""


    def execute(self, parameters, messages):
        """The source code of the tool."""
        # Clare Maher, Unity ID: cmaher2
        # GIS 540 Final Project: Reference Reach Finder Tool
        # Spring 2019

        '''''Purpose: A reference reach is a stable stream channel segment that
        serves as a model for restoration, and is described by quantitative
        field data of its morphological features (Rosgen 1998). The Reference
        Reach Finder (RRF) tool gathers information about the river basin,
        stream type, drainage area, valley slope and median grain size in
        order to find the most suitable reference reach for any project in
        North Carolina. Additionally, the RRF tool provides user flexibility
```

```
            by allowing input data from other states as well. The RRF tool is
            intended to be used by restoration designers with experience in
            natural channel design; however, several features are included to
            aid the user when input values are unknown. '''

            # Set workspace (user-defined)
            arcpy.env.workspace = parameters[0].valueAsText
            workspace = arcpy.env.workspace
            arcpy.env.overwriteOutput = True

            # Select output directory
            outputDir = parameters[1].valueAsText

            # Reference Reach Points Shapefile
            refReachPoints = parameters[2].valueAsText

            # Setting up the mapping module
            mapName = 'CURRENT'
            mxd = arcpy.mapping.MapDocument(mapName) ## 'CURRENT' or map name for Python toolbox?
            dfs = arcpy.mapping.ListDataFrames(mxd)
            df = dfs[0]

            # Adding the Reference Reach Points Shapefile to the map
            rrAddToMap = arcpy.mapping.Layer(parameters[2].valueAsText)
            arcpy.mapping.AddLayer(df, rrAddToMap)

            #Step 1: Select river basins and return reference reach points within those river basins.


            riverBasins = parameters[3].valueAsText

            rbAddToMap = arcpy.mapping.Layer(parameters[3].valueAsText)
            arcpy.mapping.AddLayer(df, rbAddToMap)

            # Field in river basins Shapefile containing river basin names.
            rbIDField = arcpy.da.SearchCursor(parameters[3].valueAsText, parameters[4].valueAsText)

            rbSelection = parameters[5].valueAsText

            rbList = rbSelection.split(';')

            for rb in rbList:
                rbQuery = "Name = '{0}'".format(rb)

            arcpy.MakeFeatureLayer_management(riverBasins, 'rbOutLayer')

            rbSelectAttr = arcpy.SelectLayerByAttribute_management('rbOutLayer', 'NEW_SELECTION', rbQu
    ery)

            arcpy.MakeFeatureLayer_management(refReachPoints, 'rrOutLayer')

            rbSelectLocation = arcpy.SelectLayerByLocation_management('rrOutLayer', 'INTERSECT', 'rbOu
    tLayer')

            arcpy.MakeFeatureLayer_management('rbOutLayer', parameters[6].valueAsText)
            arcpy.MakeFeatureLayer_management('rrOutLayer', parameters[7].valueAsText)

            rbSelectAddToMap = arcpy.mapping.Layer(parameters[6].valueAsText)
            arcpy.mapping.AddLayer(df, rbSelectAddToMap)

            rrSelectAddToMap = arcpy.mapping.Layer(parameters[7].valueAsText)
```

```
        arcpy.mapping.AddLayer(df, rrSelectAddToMap)

        rbRRPtCount = str(arcpy.GetCount_management(parameters[7].valueAsText))

        if float(rbRRPtCount) == 0:
            print "No reference reach points are within the selected river basins. Please make a new selection."
            arcpy.AddMessage("No reference reach points are within the selected river basins. Please make a new selection.")
        else:
            print "{0} reference reach points match the river basin selection criteria.".format(rbRRPtCount)
            arcpy.AddMessage("{0} reference reach points match the river basin selection criteria.".format(rbRRPtCount))

        del rbIDField


        # Step 2: Select Rosgen Stream Type (first letter) from list OR make selections to determine stream type
        rosStreamTypes = [parameters[8].values]

        streamTypeField = arcpy.da.SearchCursor(parameters[2].valueAsText, parameters[9].valueAsText)

        pickStreamType = parameters[8].valueAsText
        stList = pickStreamType.split(';')

        for st in stList:
            stQuery = "ST_Letter = '{0}'".format(st)

        arcpy.MakeFeatureLayer_management(refReachPoints, 'stOutLayer')

        stSelectAttr = arcpy.SelectLayerByAttribute_management('stOutLayer', 'NEW_SELECTION', stQuery)

        arcpy.MakeFeatureLayer_management('stOutLayer', parameters[13].valueAsText)
        stSelectAddToMap = arcpy.mapping.Layer(parameters[13].valueAsText)
        arcpy.mapping.AddLayer(df, stSelectAddToMap)

        stRRPtCount = str(arcpy.GetCount_management(parameters[13].valueAsText))

        if float(stRRPtCount) == 0:
            print "No reference reach points match the selected Rosgen Stream Type. Please make a new selection."
            arcpy.AddMessage("No reference reach points match the selected Rosgen Stream Type. Please make a new selection.")
        else:
            print "{0} reference reach points match the stream type selection criteria.".format(stRRPtCount)
            arcpy.AddMessage("{0} reference reach points match the stream type selection criteria.".format(stRRPtCount))


        # If Rosgen Stream Type (first letter) is unknown, user can select from the following three criteria to determine ST:

        entrenchment = [parameters[10].valueAsText]

        widthToDepth = [parameters[11].valueAsText]

```

```
        sinuosity = [parameters[12].valueAsText]

        del streamTypeField


        # Step 3: User enters drainage area value (square miles)
        drainageArea = float(parameters[14].valueAsText)

        drainageAreaField = arcpy.da.SearchCursor(parameters[2].valueAsText, parameters[15].valueA
sText)

        class AreaRange():
            def __init__(self, drainageArea):
                ''''' Initialize AreaRange properties '''
                self.drainageArea = float(drainageArea)
            def minMaxRange(self):
                ''''' Determine minimum and maximum drainage area values based on the user input
                value. This range will be used to search the reference reach points field containi
ng
                drainage area for sites within that acceptable range of values.'''

                if self.drainageArea < 1.0:
                    lowerBound = self.drainageArea - 0.3
                    upperBound = self.drainageArea + 0.3
                elif 1.0 <= self.drainageArea < 1.5:
                    lowerBound = self.drainageArea - 0.4
                    upperBound = self.drainageArea + 0.4
                elif 1.5 <= self.drainageArea < 3.0:
                    lowerBound = self.drainageArea - 0.5
                    upperBound = self.drainageArea + 0.5
                elif 3.0 <= self.drainageArea < 5.0:
                    lowerBound = self.drainageArea - 1.0
                    upperBound = self.drainageArea + 1.0
                elif 5.0 <= self.drainageArea < 10.0:
                    lowerBound = self.drainageArea - 2.0
                    upperBound = self.drainageArea + 2.0
                elif 10.0 <= self.drainageArea < 20.0:
                    lowerBound = self.drainageArea - 4.0
                    upperBound = self.drainageArea + 4.0

                return (lowerBound, upperBound)


        drainageRange = AreaRange(drainageArea)
        print drainageRange.minMaxRange()

        lowerBound = (drainageRange.minMaxRange()[0])
        upperBound = (drainageRange.minMaxRange()[1])

        daQuery = '{0} >= {1} AND {0} < {2}'.format(parameters[15].value, lowerBound, upperBound)


        #Making feature layers, selecting by attribute, add DA selection to the map (FOR MANUAL EN
TRY DRAINAGE AREA)
        arcpy.MakeFeatureLayer_management(refReachPoints, 'daFC')

        arcpy.SelectLayerByAttribute_management('daFC', 'NEW_SELECTION', daQuery)

        arcpy.MakeFeatureLayer_management('daFC', parameters[17].valueAsText)

        daSelection = arcpy.mapping.Layer(parameters[17].valueAsText)
```

```

        arcpy.mapping.AddLayer(df, daSelection)

        daRRPtCount = str(arcpy.GetCount_management(parameters[17].valueAsText))

        print "The tool searched for reference reaches with drainage areas in the range of {0} and {1} square miles.".format(lowerBound, upperBound)
        arcpy.AddMessage("The tool searched for reference reaches with drainage areas in the range of {0} and {1} square miles.".format(lowerBound, upperBound))

        if float(daRRPtCount) == 0:
            print "No reference reach points have a drainage area within the selected range. Please enter a new drainage area value."
            arcpy.AddMessage("No reference reach points have a drainage area within the selected range. Please enter a new drainage area value.")
        else:
            print "{0} reference reach points have a drainage area within the selected range.".format(daRRPtCount)
            arcpy.AddMessage("{0} reference reach points have a drainage area within the selected range.".format(daRRPtCount))


            # Interactive Drainage Area Calculator (Optional)
        arcpy.CheckOutExtension('Spatial')

            # User input DEM
            ##inputDEM = arcpy.GetParameterAsText()

            # Default/testing DEM:
        inputDEM = parameters[18].value

            # User input drainage point file (if they have one)
            ##drainagePtFile = arcpy.GetParameterAsText()

            # Default/testing drainage point:
        drainagePtFile = parameters[19].value


            #Watershed delineation workflow:
        fill = arcpy.sa.Fill(inputDEM)
        fill.save(workspace + parameters[20].valueAsText)

        flowDir = arcpy.sa.FlowDirection(fill)
        flowDir.save(workspace + parameters[21].valueAsText)


        flowAcc = arcpy.sa.FlowAccumulation(flowDir)
        flowAcc.save(workspace + parameters[22].valueAsText)
##        arcpy.MakeRasterLayer_management(parameters[22].valueAsText, 'flowAcc')
##        outFlowAcc = arcpy.mapping.Layer('flowAcc')
##        arcpy.mapping.AddLayer(df, outFlowAcc)

        rastCalc = arcpy.sa.Con(flowAcc, 0, 1, "Value > 100")
        rastCalc.save(workspace + parameters[23].valueAsText)
        outRastCalc = arcpy.mapping.Layer(parameters[23].valueAsText)
        arcpy.mapping.AddLayer(df, outRastCalc)

        # If not user input drainage point file, user can select drainage point on map:
        ##drainagePtSelect = arcpy.SetParameterAsText()
##        arcpy.env.extent = "MAXOF"
##        if drainagePtFile:
```

```
##        snapPourPoint = arcpy.sa.SnapPourPoint(drainagePtFile, flowAcc, 1000)
##        elif drainagePtSelect:
##            snapPourPoint = arcpy.sa.SnapPourPoint(drainagePtSelect, rastCalc, "")
##            snapPourPoint.save(outputDir + parameters[24].valueAsText)

##        snapPourPoint.save(workspace + parameters[24].valueAsText)

        # Delineate watershed
        watershedDelin = arcpy.sa.Watershed(flowDir, drainagePtFile)
        watershedDelin.save(workspace + parameters[25].valueAsText)
        #watershedRastCopy = arcpy.CopyRaster_management(watershedDelin, 'wsDelinCopy.tif')

        outWatershedPolygon = workspace + parameters[26].valueAsText

        # Convert watershed raster to watershed polygon and add to map
        watershedPoly = arcpy.RasterToPolygon_conversion(watershedDelin, outWatershedPolygon)


##        arcpy.MakeFeatureLayer_management(watershedPoly,'outWsPoly')
##        watershedAddToMap = arcpy.mapping.Layer('outWsPoly')
##        arcpy.mapping.AddLayer(df, watershedAddToMap)

        # Add area field to watershed polygon, define expression, calculate geometry of watershed 
(area in square miles)
        watershedAreaField = arcpy.AddField_management(workspace + parameters[26].valueAsText, "Ar
ea", "DOUBLE")
        calcGeomExp = "!SHAPE.AREA@SQUAREMILES!"
        areaCalc = arcpy.CalculateField_management(workspace + parameters[26].valueAsText, "Area",
 calcGeomExp, "PYTHON_9.3")


        # Create search cursor for watershed polygon area field in order to return drainage area v
alue
        wAreaSC = arcpy.da.SearchCursor(workspace + parameters[26].valueAsText, "Area")

        drainageAreaCalc = wAreaSC.next()
        arcpy.SetParameterAsText(14, drainageAreaCalc)

        # Tell the user the area of the delineated watershed
        print "The delineated watershed has an area of {0} square miles.".format(drainageAreaCalc[
0])
        arcpy.AddMessage("The delineated watershed has an area of {0} square miles.".format(draina
geAreaCalc[0]))

        # Call the AreaRange class for the drainage area calculation to determine an appropriate r
ange of search values
        drainageRangeCalc = AreaRange(drainageAreaCalc[0])
        print drainageRangeCalc.minMaxRange()

        lowerBoundCalc = abs(drainageRangeCalc.minMaxRange()[0])
        upperBoundCalc = abs(drainageRangeCalc.minMaxRange()[1])

        daQuery2 = '{0} >= {1} AND {0} < {2}'.format('Area', lowerBoundCalc, upperBoundCalc)

        #Make feature layers, select by attribute within reference reach feature class, add final 
DA selection to the map
        arcpy.MakeFeatureLayer_management(outWatershedPolygon, 'daFCCalc')

        arcpy.SelectLayerByAttribute_management('daFCCalc', 'NEW_SELECTION', daQuery2)

        arcpy.MakeFeatureLayer_management('daFCCalc', parameters[28].valueAsText)
```

```
        daSelectionCalc = arcpy.mapping.Layer(parameters[28].valueAsText)

        daSelectionCalc.transparency = 50

        arcpy.mapping.AddLayer(df, daSelectionCalc)

        # Tell user the range of drainage area values used in reference reach selection
        print "The tool searched for reference reaches with drainage areas in the range of {0} and {1} square miles.".format(lowerBoundCalc, upperBoundCalc)
        arcpy.AddMessage("The tool searched for reference reaches with drainage areas in the range of {0} and {1} square miles.".format(lowerBoundCalc, upperBoundCalc))

        # Determine number of reference reaches found based on search criteria, and notify user of the results.
        daRRPtCount2 = str(arcpy.GetCount_management(parameters[28].valueAsText))

        if float(daRRPtCount2) == 0:
            print "No reference reach points have a drainage area within the selected range. Please enter a new drainage area value."
            arcpy.AddMessage("No reference reach points have a drainage area within the selected range. Please enter a new drainage area value.")
        else:
            print "{0} reference reach points have drainage areas within the selected range.".format(daRRPtCount2)
            arcpy.AddMessage("{0} reference reach points have drainage areas within the selected range.".format(daRRPtCount2))


        del wAreaSC
        del drainageAreaField



        # Step 4: Valley Slope
        valleySlope = float(parameters[29].valueAsText)
        valleySlopeField = arcpy.da.SearchCursor(parameters[2].valueAsText, parameters[30].valueAsText)

        def ValleySlopeRange(valleySlope):
            tolerance = float(valleySlope * 0.05)
            plusTolerance = float(valleySlope + tolerance)
            minusTolerance = float(valleySlope - tolerance)
            return (minusTolerance, plusTolerance)

        slopeSelectionRange = ValleySlopeRange(valleySlope)
        print slopeSelectionRange

        minusTolerance = (slopeSelectionRange[0])
        plusTolerance = (slopeSelectionRange[1])

        vsQuery = '{0} >= {1} AND {0} < {2}'.format(parameters[30].value, minusTolerance, plusTolerance)

        arcpy.MakeFeatureLayer_management(refReachPoints, 'vsFC')

        arcpy.SelectLayerByAttribute_management('vsFC', 'NEW_SELECTION', vsQuery)

        arcpy.MakeFeatureLayer_management('vsFC', parameters[32].valueAsText)

        vsSelection = arcpy.mapping.Layer(parameters[32].valueAsText)
```

```
        arcpy.mapping.AddLayer(df, vsSelection)

        print "The tool searched for reference reaches with valley slopes in the range of {0} and {1} ft/ft.".format(minusTolerance, plusTolerance)
        arcpy.AddMessage("The tool searched for reference reaches with valley slopes in the range of {0} and {1} ft/ft.".format(minusTolerance, plusTolerance))

        vsRRPtCount = str(arcpy.GetCount_management(vsSelection))
        #print vsRRPtCount

        if float(vsRRPtCount) == 0:
            print "No reference reach points have a valley slope within the selected range. Please enter a new valley slope value."
            arcpy.AddMessage("No reference reach points have a valley slope within the selected range. Please enter a new valley slope value.")
        else:
            print "{0} reference reach points have a valley slope within the selected range.".format(vsRRPtCount)
            arcpy.AddMessage("{0} reference reach points have a valley slope within the selected range.".format(vsRRPtCount))



        # Interactive valley slope calculator: If valley slope unknown

        contours = parameters[33].value

            # Add contours layer to map if user chooses interactive valley slope calculator
        contoursAddToMap = arcpy.mapping.Layer(parameters[33].valueAsText)
        arcpy.mapping.AddLayer(df, contoursAddToMap)

            # Field that shows elevation (feet) of contour lines
        elevationField = arcpy.da.SearchCursor(parameters[33].valueAsText, parameters[34].valueAsText)

            # Display elevation ('Z_Feet') labels on map for user to view
        contourLayer = arcpy.mapping.ListLayers(mxd, contoursAddToMap)[0]

        if contourLayer.supports("LABELCLASSES"):
            for lblClass in contourLayer.labelClasses:
                lblClass.showClassLabels = True
        lblClass.expression = "[Z_Feet]"
        contourLayer.showLabels = True

            # Did not have enough time to finsih figuring out how to allow user to create points/draw lines on map
            # After doing some research, I'm not sure if that's even a possibility?
        ##createValleyFC = arcpy.CreateFeatureclass_management('/output/', 'createValleyFC', 'POINT')
        ##arcpy.AddField_management(createValleyFC, 'v_length', 'DOUBLE')
        ##
        ### Create insert cursor to insert new rows in createValleyFC attribute table for points selected by user
        ##valleyIC = arcpy.da.InsertCursor(createValleyFc, ("SHAPE@XY"))

        ##createValleyLineOut = arcpy.mapping.Layer(createValleyFC)
        ##arcpy.mapping.AddLayer(df, createValleyLineOut)

        ##valleyLengthCalc = arcpy.CalculateField_management(createValleyFC, 'v_length', "!SHAPE.LENGTH@FEET!", "PYTHON_9.3")
```

```
        ##vLengthSC = arcpy.da.SearchCursor(createValleyFC, 'v_Length')
        ##valleyLength = vLengthSC.next()

        valleyElev1 = float(parameters[35].value)
        valleyElev2 = float(parameters[36].value)
        valleyLength = float(parameters[37].value)

        def ValleySlopeCalculator(valleyElev1, valleyElev2, valleyLength):
            vSlopeCalc = abs(valleyElev1 - valleyElev2)/valleyLength
            return (vSlopeCalc)

        vSlopeCalc = ValleySlopeCalculator(valleyElev1, valleyElev2, valleyLength)


        # How to get value to appear to user? SPAT is for getting text from derived output
        arcpy.SetParameterAsText(38, vSlopeCalc)

        vSlopeCalcRange = ValleySlopeRange(vSlopeCalc)
        print vSlopeCalcRange

        minusToleranceCalc = (vSlopeCalcRange[0])
        plusToleranceCalc = (vSlopeCalcRange[1])

        vsQuery = '{0} >= {1} AND {0} < {2}'.format('V_Slope', minusToleranceCalc, plusToleranceCa
lc)

        arcpy.MakeFeatureLayer_management(refReachPoints, 'vsCalc')

        arcpy.SelectLayerByAttribute_management('vsCalc', 'NEW_SELECTION', vsQuery)

        arcpy.MakeFeatureLayer_management('vsCalc', parameters[40].valueAsText)

        vsCalcSelection = arcpy.mapping.Layer(parameters[40].valueAsText)

        arcpy.mapping.AddLayer(df, vsCalcSelection)

        vsCalcRRPtCount = str(arcpy.GetCount_management(vsCalcSelection))
        #print vsCalcRRPtCount


        if float(vsCalcRRPtCount) == 0:
            print "No reference reach points have a valley slope within the selected range. Please
 enter a new valley slope value."
            arcpy.AddMessage("No reference reach points have a valley slope within the selected ra
nge. Please enter a new valley slope value.")
        else:
            print "{0} reference reach points have a valley slope within the selected range.".form
at(vsCalcRRPtCount)
            arcpy.AddMessage("{0} reference reach points have a valley slope within the selected r
ange.".format(vsCalcRRPtCount))
            arcpy.mapping.AddLayer(df, vsCalcSelection)

        del valleySlopeField
        del elevationField
        ##del vLengthSC

        # Step 5: Select the median grain size (D50) of the project site
        medParticleSize =  [parameters[41].valueAsText]

        medParticleSizeField = arcpy.da.SearchCursor(parameters[2].valueAsText, parameters[42].val
ueAsText)
```

```

        #medParticleSizeSelect = parameters[41].values  #new variable for selection from list? probably not.

        d50Query = "{0} = '{1}'".format('D50_Class', parameters[41].valueAsText)

        arcpy.MakeFeatureLayer_management(parameters[2].valueAsText, 'd50FC')

        arcpy.SelectLayerByAttribute_management('d50FC', 'NEW_SELECTION', d50Query)

        arcpy.MakeFeatureLayer_management('d50FC', parameters[43].valueAsText)

        d50Selection = arcpy.mapping.Layer(parameters[43].valueAsText)
        arcpy.mapping.AddLayer(df, d50Selection)

        d50RRPtCount = str(arcpy.GetCount_management(d50Selection))

        if float(d50RRPtCount) == 0:
            print "No reference reach points have a median grain size (D50) within the selected range. Please choose a new D50 value."
            arcpy.AddMessage("No reference reach points have a median grain size (D50) within the selected range. Please enter a new D50 value.")
        else:
            print "{0} reference reach points have a median grain size (D50) within the selected range.".format(d50RRPtCount)
            arcpy.AddMessage("{0} reference reach points have a median grain size (D50) within the selected range.".format(d50RRPtCount))

        del medParticleSizeField

        refReachFinalOutput = parameters[44].valueAsText

        # This merge and delete identical worked perfectly find when executed in Python Win, but I have yet to be successful with it through the Python Toolbox script.
        refReachSelectionMerge = arcpy.Merge_management([rrSelectAddToMap, stSelectAddToMap, vsSelection, vsCalcSelection, daSelection, d50Selection], refReachFinalOutput)
##        refReachSelectionMerge = arcpy.Merge_management([parameters[6].valueAsText, parameters[13].valueAsText, parameters[32].valueAsText, parameters[43].valueAsText, parameters[17].valueAsText, parameters[40].valueAsText, parameters[28].valueAsText], refReachFinalOutput)

        deleteIdentical = arcpy.DeleteIdentical_management(refReachSelectionMerge, 'Name')


##        arcpy.MakeFeatureLayer_management(deleteIdentical, parameters[45].valueAsText)
##
##        refReachFinalOutputLayer = arcpy.mapping.Layer(parameters[45].valueAsText)
##
##        refReachFinalOutputLayer.symbology
##
##        arcpy.mapping.AddLayer(df, refReachFinalOutputLayer)

        # Saving a copy of the map (when running in IDE)
##        copyMap1 = workspace + '/' + mapName[:-4] + '_Copy.mxd'
##        mxd.saveACopy(copyMap1)

        # Delete map document object
        del mxd


        return
```