A Network-based Model for Quantifying Regional-Scale Nitrogen Transport within River Basin Systems

Jonathan L. Goodall, A.M.ASCE[1], John P. Fay[2], and David L. Bollinger, Jr.[3]

Abstract

A network-based model for quantifying nitrogen fate and transport at the regional-scale is presented as a tool for aiding decision makers in managing river basin systems. The model is based on the United States Geological Survey (USGS) SPAtially Referenced Regression on Watershed attributes (SPARROW) model and implements the prediction portion of SPARROW for quantifying overland and in-stream losses of nitrogen within a river basin system. The intention of the model is to serve as an Application Programmers Interface (API), or a software library, that can be used to construct analysis and visualization tools that address a variety of management questions. By leveraging the model API along with other existing open source and freely available software libraries for visualization, numerical methods, and network algorithms, it is possible to quickly produce decision support tools that relate pollution sources to downstream loadings. These tools can be used by managers in order to understand how hypothetical regional-scale changes to the landscape (e.g. development, climate change, etc.) might impact water quality conditions. The use of the model API is demonstrated with a case study that addresses three potential management questions regarding nitrogen fate and transport in the Upper Neuse River Basin of North Carolina.

Subject Headings: Water quality modeling; Nonpoint pollution; Software design; Decision support systems

[1] Assistant Professor, Department of Civil and Environmental Engineering, University of South Carolina, 300 Main Street, Columbia, SC 29208, goodall@engr.sc.edu

[2] Research Associate, Nicholas School of the Environment, Box 90328, Duke University, Durham, North Carolina, 27708, john.fay@duke.edu

[3] Staff Scientist, Sher Leff LLP, 450 Mission Street, Ste. 400, San Francisco, California, 94105, dave.bollinger@alumni.duke.edu

1 Introduction

River basin management requires a variety of modeling approaches to most efficiently address existing or potential problems. These modeling techniques can be broadly defined along a continuum from those that make use of physically-based mechanisms to those that rely on empirical relationships to characterize system response (Schwartz et al. 2006). It is common practice for watershed modelers or watershed managers to select a modeling technique from along this continuum based on many different criteria including, but not limited to, the management question at hand, the data available for describing the system, and the resources available to develop a model of the system.

Physically-based models (e.g. HSPF and SWAT) may be chosen if detailed questions are to be addressed about the fate and transport of a contaminant in space and time. Such models include sophisticated mathematical representations of system dynamics that implement many of the known processes that influence mass transport within a watershed system, and they therefore are able to provide predictions at fine spatial and temporal scale. However, the cost associated with parameterizing and calibrating this class of models for a specific watershed can be excessive.

A watershed modeler, on the other hand, may select an empirical modeling approach if only long term average loading estimates are needed, or if resources are not available to develop a physically-based model. Empirical modeling concepts such as export coefficients estimate steady-state mass loadings to waterbodies based on the land use types in that waterbody's watershed (Johnes 1996). While conceptually simple and inexpensive to implement, purely empirical models do not account for physical processes known to occur as mass moves through a watershed system (e.g. losses associated with overland and in-stream mass transport), and thus such approaches may not accurately predict mass loading rates.

Other modeling techniques can be placed between these extremes of the continuum because they borrow ideas from both physically and empirically-based models. A good example of such a model is the USGS SPAtially Referenced Regression On Watershed attributes (SPARROW) model (Smith et al. 1997; Schwarz et al. 2006). SPARROW is physically-based in that it includes representations for mass transport from the land-surface to the stream network, and from one location along the stream network to a downstream location in the stream network. It is empirical in that it does not include a complete physical representation of the watershed, but instead relies on regression techniques to determine optimal model coefficients that minimize the difference between observed and predicted mass loading rates.

Therefore, highlighting the benefits of the SPARROW model, one can say that it provides more scientific rigor compared to models based purely on empirical concepts, but is less expensive to implement compared to physically based models. The benefits come at a loss, namely that the model is more difficult to set up and run compared to purely empirical approaches, and it does not provide the spatial and temporal resolution possible with a physically-based modeling approach. However, SPARROW represents an important and, in our opinion, underutilized model within the set of tools available to watershed and water resource managers.

While SPARROW might be an underutilized tool for watershed and water resources management, it is a well tested and documented model. SPARROW has been applied to numerous spatial regions to address water quality questions primarily related to nutrient transport, although some preliminary work has been completed on modeling sediment transport using SPARROW (Schwarz 2008). Smith et al. (1997) used 414 monitoring stations to determine long-term average transport of total nitrogen and total phosphorous for approximately 60,000 river reaches within the contiguous United States. More localized studies have also been conducted that have calibrated SPARROW for New England (Moore et al. 2004), the Chesapeake Bay Watershed (Brakebill and Preston 2003; Brakebill and Preston 1999; Preston and Brakebill 1999), the Mississippi River Basin (Alexander et al. 2000; Alexander et al. 2008) and major river basins within North Carolina (McMahon et al. 2003).

The process of applying the SPARROW model can be considered as the combination of a calibration phase and a prediction phase. In the calibration phase, model coefficients are determined using a least squares error technique that minimizes the difference between model predictions and observed water quality conditions (Schwarz et al. 2006). The prediction phase of SPARROW then uses these model coefficients to estimate mass loadings for all reaches within the river network. The result of the calibration phase, therefore, is a set of model coefficients that describe the fate and transport of mass through the river basin system. The result of the prediction phase is a quantification of the mass loadings for each reach within the river basin, along with loss rates that describe how upstream watershed properties impact downstream water quality conditions. Once a model has been calibrated, it is possible to make hypothetical changes to current watershed properties to estimate their potential impact on downstream water quality conditions. This can aid watershed managers in understanding the potential impact of changes to the watershed, and in testing strategies to minimize impacts through the adoption of best management practices (BMPs).

We argue that, by separating the calibration phase of SPARROW from the prediction phase, water managers can more easily benefit from tools that implement the prediction portion of SPARROW to understand regional-scale nitrogen transport. Watershed managers could make use of published model coefficients and free, open source programming languages to predict water quality conditions under both current and proposed conditions. The current implementation of SPARROW does include a both calibration and prediction capabilities, however its current implementation uses the SAS® statistical programming language, which limits it's utility as a tool for use by watershed managers. Furthermore, writing the prediction algorithms in SPARROW within an object-oriented structure makes it easier to apply the algorithms to a broader range of potential watershed management questions. This approach, therefore, would benefit the community by providing a freely-available, object-oriented software library that fills the gap between the physically and empirically-based models available now for addressing water quality concerns.

To this end, we present an implementation of the prediction portion of SPARROW that is intended to serve as an Application Programming Interface (API) for constructing decision support tools. An API is a software library that can be used to create one or more applications (Eck 2006). APIs are ubiquitous in software engineering and examples range from the Windows API that allows software programs to interact with the Windows operating system, to the Google Maps API that provides a way for third-party developers to quickly create online maps by leveraging Google Technology. The SPARROW API presented in this paper follows this same approach by providing a higher-level programming library that provides an easier way to incorporate SPARROW prediction routines within watershed decision support tools.

The SPARROW API is written in the Python programming language and, as is the convention with Python tools, we have named the API pySPARROW. Python is a free, open source, object-oriented, and platform independent programming language that is becoming increasingly popular in many disciplines from software engineering to science (Oliphant 2007). PySPARROW makes use of many Python packages to handle lower-level file management, network flow algorithms, numerical analysis, and visualization. These packages include GraphViz for visualization, NetworkX for network (or graph) processing, NumPy for numerical analysis, and pyTables for file input and output. All of these libraries are free and open source. We made the library independently of any commercial software so that it can be freely distributed to users.

The following two sections describe the design and implementation of the pySPARROW API. Section 2 provides the mathematical background to the prediction phase of SPARROW and Section 3 presents how the mathematics were implemented within an object-oriented software design. Section 4 discusses a case study where pySPARROW is applied to address three management questions regarding fate and transport of Nitrogen in the Upper Neuse River Basin.

2 Model Description

SPARROW is a nonlinear, multivariate regression model that relates spatially referenced watershed attributes to observed water quality conditions (Smith et al, 1997; Schwarz, 2006). If one considers a river basin network consisting of a set of reaches *J*, then SPARROW states that the mass loading to one of those reaches *i* can be estimated as the sum of the mass loadings from all contributing reaches $j \in J(i)$ for all pollution sources $n=1..N$ (Equation 1).

$$L_i = \sum_{n=1}^{N} \sum_{j \in J(i)} s_{n,j} D(Z_j) K(T_{i,j}) \tag{1}$$

The pollution source *n* contributed by reach *j*, $s_{n,j}$, is transported to downstream reach *i* according to two nonlinear processes (Figure 1). The first process, represented in Equation 1 as $D(Z_j)$, describes the transport from the land surface to the stream and is a function of the watershed properties for reach *j*, Z_j. The second process, represented in Equation 1 as $K(T_{i,j})$, describes in-stream transport from reach *j* to reach *i* and is a function of the flow path properties between reaches *i* and *j*.

For nonpoint source pollution, the default setting in SPARROW is to assume that land surface to stream transport follows a first order decay process

$$D(Z_j) = \beta_n e^{-\alpha' Z_k} \tag{2}$$

where β_n and α are regression-derived coefficients. For point sources, $D(Z_j)=1$ signifying that there is no land-to-stream decay for point source pollution. Observed loads at monitoring stations are treated as boundary conditions for determining the loading to a downstream reach. Therefore, when calculating the loading to reach *i*, contributing reaches are those upstream of the reach and downstream of any monitoring stations above that reach. This is expressed in the model by letting $D(Z_j)=1$ for observed upstream loadings.

The in-stream transport term, $K(T_{i,j})$, is also expressed as a first-order decay process

$$K(T_{i,j}) = e^{-\delta' T_{i,j}} \tag{3}$$

where δ is a regression-related coefficient that is a function of streamflow. Low-flow streams (typically taken as less than 500 cubic feet per second) are considered as one class, medium-flow streams (typically taken as between 500 and 10,000 cubic feet per second) are considered as a second class, and high-flow streams (typically greater than 10,000 cubic feet per second) are considered as a third class of streams. The regression coefficient δ is determined for each stream flow class. Finally, for reservoirs an additional loss rate is considered to account for settling within reservoirs. Therefore, the loss of mass within reservoirs is modeled as the product of a first-order decay rate and a loss associated with a settling rate. This is expressed in Equation 4 which states

$$K(T_{i,j}) = e^{-\delta' T_{i,j}} \left[\frac{1}{1 + \theta_i / h_{load,j}} \right] \tag{4}$$

where θ_i is a regression-derived coefficient and $h_{load,i}$ is the areal hydraulic load (length per time) of the reservoir.

SPARROW requires information about the river basin properties for parameterization. For this study, we made us of an existing SPARROW dataset provided with version 2.8 of the SPARROW model (Schwarz 2008). The dataset consists of a river network with all required attributes, source terms, and observed loading rates for the contiguous United States at a 1:500,000 scale. (Nolan et al. 2003). The motivation for using this database was to verify that pySPARROW reproduced predictions made by the original and tested SPARROW prediction routines implemented in original SPARROW model.

Finally, while this paper makes use of the default SPARROW model settings, it is important to note the flexibilities inherit in the SPARROW model that could also be implemented within future versions of pySPARROW. The model allows for readjustment of the mathematical representations for land-to-stream, in-stream, and reservoir loss rates. It is possible to make use of finer-scale stream networks (e.g. the 1:100K National Hydrography Dataset), assuming that data is available at that scale for parameterizing the model. It is also possible to consider additional sources of nitrogen within the river system, as well as other watershed and stream properties that may influence the fate and transport of nitrogen within the system. However, any changes to the basic model structure or parameterization would require a recalibration of the model using the original SPARROW implementation. PySPARROW does not perform calibration; it only implements the prediction algorithms to understand fate and transport within a particular system. Thus, pySPARROW is intended to be a supplemental tool used alongside the original SPARROW model and is not intended to be a replacement for the original SPARROW model.

3 Model Implementation

The basic mathematics of SPARROW prediction represented in Section 2 were implemented in pySPARROW as five object classes: (1) Reach, (2) Waterbody, (3) Network, (4) Database, and (5) Utility (Figure 2). Each class represents either some real-world object or system object needed to implement the prediction routines. The following section provides details on each of the pySPARROW classes. The implementation of pySPARROW has been tested against the national SPARROW model to insure that pySPARROW is able to reproduce the prediction results in the original version of SPARROW.

3.1 The Reach Class

The reach class is the basic modeling unit in SPARROW. It represents some length of the overall river network, most often the portion of stream between river confluences. Each reach has an associated catchment that represents the land surface area contributing mass to that reach. Because a one-to-one relationship is assumed between stream reaches and watersheds, the properties of a watershed are directly associated to the reach within that catchment. It follows that each watershed is a lumped unit within the model structure that acts as a source of mass to the system. The scale at which one defines reaches and watersheds is left to the user.

Each reach object must have an identification number that is unique over the entire network. This concept is expressed in all hydrography datasets including the National Hydrography Dataset (NHD), the Reach File datasets, and the Arc Hydro data model (Maidment 2002). The remaining properties (Table 1) are used to quantify the fate and transport of nitrogen within the river basin. These include the nitrogen sources associated with the reach (s_j) catchment properties associated with the reach (Z_j), and the stream properties associated with the reach (Q_j, t_j).

The Reach class includes methods for estimating the initial mass (get_source) and land-to-water delivery of mass for each pollution source (get_runoff) in the model. The get_source method returns the product of the source term for the reach j and the beta coefficient ($\beta_n s_{n,j} \ \forall n \in N$). The get_runoff method is actually a combination of methods that return the mass flux from the land surface to the stream for a specific pollution source ($s_{n,j} D(Z_j) \forall n \in N$), or for all sources collectively ($\sum_{n=1}^{N} s_{n,j} D(Z_j)$). The methods currently implement the following pollution sources: point sources, agricultural waste nonpoint sources, fertilizer nonpoint sources, atmospheric deposition nonpoint sources, and sources associated with land use and land cover types urban, shrub, grass, transitional, and forest.

The object-oriented structure of pySPARROW allows each Reach object to calculate a watershed-to-stream loading based on its current properties. Changes to these properties will immediately be reflected in the loading values calculated within the Reach object methods. This allows for a flexible analysis environment where a modeler might define proposed or potential changes to in-memory representations of the stream network in order to analyze how those changes impact nitrogen fate and transport within the system. The pySPARROW API handles this and other low-level operations (e.g. data input and output and network flow algorithms) so that such analysis can be developed much more efficiently.

3.2 The Waterbody Class

A waterbody class is used to compute loadings from upstream reaches given the river network topology. A waterbody, by using the network object, knows its upstream reaches, and so a waterbody object can determine the mass loading delivered by these upstream reaches for each nitrogen source. The algorithm for the loading methods is to first use the Network's get_upstream_reaches method (discussed in the next section) to identify all upstream reaches for that location of the network. Then for each reach in this list of upstream reaches, that reach's land-to-stream flux is calculated using the runoff method (get_ runoff). Finally, the in-stream loss due to transport from the source reach to the designation waterbody is estimated by first using the Network to identify the path from the source reach to the designation waterbody, and then, for each reach along the path, using that reach's properties to estimate both a reach and reservoir loss rate. These loss rates are accumulated over the entire travel path to calculate a total in-stream loss rate from source to designation.

3.3 The Network Class

A network class contains the topology information between reaches within the river network. Each node within the network represents a reach object, and each edge represents the connection between reaches. Using the network's connectivity information, it is possible to derive flow paths between any two reaches within the network, find all reaches upstream for a given reach, and query for the next downstream or next upstream reach for any reach within the network. These network algorithms are used to calculate mass transport through the in-stream system, as described in the Waterbody Class section. They can also be used more generally to analyze and understand hydrologic connectivity as well as the fate and transport of nitrogen within the river basin system.

A network object has six methods. First, it includes a method to query for any reach within the network, given that reach's unique identifier (get_reach). There is also a method for accessing a waterbody object (get_waterbody). A network class also includes methods for traversing the network that are used in load calculations in the waterbody object, or can be used more generally for analyzing network connectivity. Methods exist for identifying all reaches upstream of an outlet reach (get_upstream_reaches), all reaches along a flow path provided a starting and ending reach (get_path), and the next downstream or upstream reaches (get_next_upstream_reaches and get_next_downstream_reaches).

Many of these methods are built on an existing open source and freely available graph API named NetworkX (Hagberg et al. 2008). NetworkX is being developed at Los Alamos National Laboratory as a tool for scientists and engineers to explore and analyze network data structures. We choose to leverage NetworkX in pySPARROW because it has been optimized to handle large graphs and includes many algorithms for analyzing graph structures. When a new network object is initialized using pySPARROW, the user provides an input file that stores the network and reach attributes, and internally pySPARROW builds a NetworkX digraph object to represent the river system. In the Network class of PySPARROW, each method calls a specific function or method contained in NetworkX to perform the network operation. For example, the get_upstream_reaches method of the Network class calls NetworkX function single_source_shortest_path using the river network structure with the flow directions reversed.

This design approach ensures that the pySPARROW methods are optimized to handle large networks and allows for a more complete end tool.

3.4 The Database Class

The final two classes, the database class and the utility class, do not represent real world entities but are instead designed to be support classes in pySPARROW. The database class facilitates data input and output (I/O). It includes methods to add, remove, and update reaches within the database file. The file format used to store river network properties is the Hierarchical Data Format (HDF), a binary file format developed at the National Center for Supercomputing Applications (NCSA), to reduce file size and improve reach and search performance for large graphs. The PyTables Python library is used as an API for reading and writing to the HDF file. This provides a free, cross-platform solution for managing data within pySPARROW.

The precise attributes contained within a pySPARROW database are listed in Table 1. These fields represent the attribute for reach objects including the concept of a *ToUID* that links each reach to its downstream neighbor. The Database object also includes the idea of versioning so that one database can contain multiple instances of the same river network system with modifications to the reach attributes (e.g. different land use distributions). When one is creating a new network object, a version number can be specified to create a particular instance of the network. The purpose of versioning is to allow for persistent storage of particular river network configurations that may be of interest to water managers. It should be noted that once a network object is created from the database, it is also possible to simply change the reach attributes of that in-memory network dataset without the need to write the network to file. Thus, it is possible to efficiently test various network configurations without the performance cost of file input and output (provided that the computer has sufficient RAM for storing all network configurations).

3.5 The Utility Class

Finally, the utility class is used to store general functions that users are most likely to extend within their own applications of pySPARROW. For example, the calculation for in-stream loss rates is stored in the utility class because this calculation can be modified in the core SPARROW model, which requires that it can also be modified in pySPARROW. The Utility class stores the SPARROW calibration coefficients for this same reason. Other methods may be added to the Utility class including tools to transform data between a comma separated values file and a HDF file, methods to create general visualizations of networks, etc. The general purpose design of this class, therefore, is as a place to store useful routines that do not apply specifically to one of the other four classes in pySPARROW.

4 Case Study Application

The previous sections covered the mathematical background and computer software implementation of pySPARROW. This section describes a simple case study to demonstrate how pySPARROW could be used to support decision making. The focus of the case study is the Upper Neuse River Basin which drains the majority of the Research Triangle Park area in central North Carolina (Figure 3a). Taking this river basin as a case study, consider the following simple questions that a watershed manager might ask about water quality conditions within the basin.

(1) What is the incremental, per unit area loading for each reach within the Upper Neuse, and how do these loadings vary for different nitrogen sources?

(2) What portion of the incremental load from each watershed is transported to the river basin outlet?

(3) How would a 10% increase in fertilizer application impact loading at the watershed outlet?

Each question can be analyzed using pySPARROW, and we present the steps required to perform the analysis within this section. We used the National SPARROW database and model calibration available from the USGS to address each question (Schwarz et al. 2006). Therefore, this analysis could be done just as easily for any river within the contiguous United States. The database includes nitrogen sources such as point sources from municipal water, fertilizer application sources, atmospheric deposition, and contributions from urban land use types and is implemented at a 1:500,000 spatial scale using the River Reach File dataset produced by the U.S. Environmental Protection Agency.

To visualize the river basin, we leveraged a network drawing algorithm called *neato* which is provided as part of the Graph Visualization Software (GraphViz) developed by AT&T Research Labs for visualizing large, complex networks (Ellson et al. 2002). Figure 3 presents the network in (a) geographic space and (b) drawn using the GraphViz *neato* algorithm. The advantage of this network representation compared to a geographic view is that it more clearly highlights the tributaries flowing into the Neuse River. This can be a helpful way to visualize large, complex river basin systems, particularly at finer spatial scale when there are hundreds or thousands of catchments within the river basin.

4.1 Question 1: What is the incremental, per unit area loading for each reach within the Upper Neuse, and how do these loadings vary for different nitrogen sources?

This first question demonstrates the land-to-river transport process in SPARROW. The basic algorithm is to iterate over all reaches within the river basin and, for each reach, use that reach's get_ runoff method to estimate the total incremental load delivered from that reach's watershed to the stream within that watershed. The results (Figure 4a) show which reaches contribute the highest amount of nitrogen to the stream channel, after normalizing by the contributing area, by using a graduated symbology where the size and color of the node represents the magnitude of the loading delivered by that reach to the stream channel. This visualization makes it clear which reaches are contributing disproportional loading to the river network. A water resource manager might target his or her efforts on these watershed units to reduce nitrogen loading to the river network system.

The previous algorithm can be extended to estimate how each source of nitrogen is contributing incremental, per unit area loadings to the stream network. These various loading components can be calculated by calling the get_ runoff method for each pollution source. The result of this analysis (Figure 4b) reveals that fertilizer contributes the highest loading to the stream, and that interesting spatial patterns exist in the loading for different pollution sources to the stream. For example, the amount of animal waste delivered to the main branch by some tributaries within the river basin greatly exceeds the load contributed by other tributaries. This information might allow a water resource manager to better target nutrient reduction measures on specific tributaries and identify source-specific reduction strategies to improve water quality within the watershed.

4.2 Question 2: What portion of the incremental load from each watershed is transported to the river basin outlet?

In the previous question, there was no consideration of in-stream losses associated with mass transport through the river network. However, this question does require an analysis of the system that includes load reductions due to in-stream transport. The pySPARROW Network object can be used to calculate the in-stream losses resulting from transport along some length of the river network. To answer the specific question posed, the algorithm would be to iterate through all reaches in the network and, using the Network object, estimate the reduction in a reach's incremental loading after it has been transported to the river basin outlet. This reduction is then multiplied by the reach's runoff loading to estimate the amount of nitrogen delivered to the basin outlet. The results can be visualized by symbolizing each node based on the amount of loading that reaches the basin outlet (Figure 5). The results illustrate the fact that reduction is not simply a function of linear distance in SPARROW, but instead it is a function of travel time and streamflow rate. A water resource manager can use such information to identify and target the most significant contributors to a waterbody, a question that is relevant to the Total Maximum Daily Load (TMDL) program.

4.3 Question 3: How would a 10% increase in fertilizer application impact loading at the watershed outlet?

The final question demonstrates how pySPARROW could be used to understand the potential impacts to water quality caused by changes in the river basin system. As an example, one could address this specific question by setting the fertilizer attribute of each reach within the network to a value 10% higher than its original value, and then estimating the total loading for any location along the river network by using a Waterbody object to accumulate upstream loads. Comparing the total loading estimations for the unchanged network with the estimates for the network with a 10% increase in fertilizer loads for seven locations along the stream network (Figure 6) shows the potential impact on the river basin system to the increased fertilizer input. This example also demonstrates the idea that a waterbody object is not limited to being placed at the river basin outlet. A waterbody object can be placed at any reach in the network to quantify loading to that reach from upstream reaches in the network.

5 Summary, Discussion, and Future Work

The fundamental problem that this work attempts to address is the need for watershed prediction models that balance the benefits and limitations of physically and empirically based modeling approaches. Physically-based models are expensive to parameterize and calibrate, but they can offer detailed information to water resources managers that cannot be produced with empirical models. SPARROW is unique in that it incorporates principles from both physically-based and empirical models. For example, it includes two nonlinear terms to represent overland and in-stream transport mechanisms, respectively. However, both terms are greatly simplified representations of the true physics that govern nutrient fate and transport within a riverine system, and parameterization of the terms is achieved with regression instead of physically-derived parameters. This balance between physical and empirical positions SPARROW well as a "first-cut" model for accessing general trends in water quality at a regional scale. Thus, the goal of the research was to reengineer the prediction portion of the SPARROW model as an object-oriented software library that could be leveraged within decision support systems. In software engineering terminology, pySARROW is an Application Programming Interface (API) in that it provides a collection of objects that are used to create a software system.

PySPARROW is demonstrated through a case study that addresses three hypothetical management questions. Each question was addressed by writing a short script that leveraged pySPARROW to calculate properties of nitrogen transport within the Upper Neuse River Basin. This simple case study highlights the utility of pySPARROW for quickly creating tools that address a specific management question, and more generally it demonstrates the power of high-level scripting languages as a powerful and transparent analysis environment for water resources management. End users and stakeholders could make use of output visualizations resulting from these scripts. It is also possible, although it was not demonstrated here, to use the pySPARROW API as a utility within a Graphical User Interface (GUI)-based decision support system. This would allow managers and stakeholders the ability to adjust parameters of an analysis and receive more immediate feedback through the GUI.

We believe that writing models as APIs within a high-level scripting language is an attractive option because it allows decision support system developers the ability to combine software libraries in creative ways to create useful end products. Each environmental problem is unique either in terms of the location, system abstraction, or contaminate of concern. By providing analysts with software libraries instead of prebuilt software applications, the analyst is better able to combine the libraries in a specific way to help address the question at hand. Increasing the number and variety of APIs that use these higher level languages will provide new tools for creating decision support systems that aid watershed managers. We believe that such tools will ultimately lead to more effective use of resources in addressing water quality concerns.

This concept is demonstrated with pySPARROW itself. PySPARROW makes use of many existing APIs created and maintained by external organizations and offered as free, open source resources in order to achieve its desired outcome. Most notably is the API for network flow analysis called NetworkX (Hagberg, 2008) that provides highly optimized algorithms for determining network flow. Other APIs were also leveraged for creating network visualization, for data input and output, and for linear algebra. This dependence on others' software does create some fragility in the end product, but because each package is open source and freely available, it is possible to bundle all dependencies into one complete solution so that if one package is discontinued, it does not break the dependent application. Most water resources models are not developed as APIs, but instead as standalone applications. However, the API approach offers tremendous advantages for quickly producing models that can be combined into applications in order to address environmental problems.

The development of the pySPARROW API will continue and efforts will be made to provide additional functionality to water resource managers. For example, future work will be directed toward extending the current pySPARROW implementation to consider uncertainty estimates produced by SPARROW (Robertson et al. 2009). While the SPARROW model performs a boot-strap analysis to estimate the probability distributions of model coefficients (Schwarz et al. 2006), the current implementation of pySPARROW ignores these probability distributions and only uses mean coefficient estimates in the loading predictions. Uncertainty is a critical component of water resources management, and so this limitation of pySPARROW will be addressed in forthcoming versions of the model.

PySPARROW and the example scripts presented in this paper are offered free of charge and open source from http://code.google.com/p/pysparrow.

Acknowledgments

The authors wish to acknowledge Anne Hoos of USGS for her help with implementing the SPARROW model prediction logic within pySPARROW.

References

Alexander, R. B., Smith, R. A., and Schwarz, G. E., 2000. Effect of stream channel size on the delivery of nitrogen to the Gulf of Mexico. Nature 403(6771) 758-761.

Alexander, R. B., Smith, R. A., Schwarz, G. E., Boyer, E. W., Nolan, J. V., and Brakebill, J. W., 2008. Differences in phosphorus and nitrogen delivery to the gulf of Mexico from the Mississippi river basin. Environmental Science & Technology 42(3) 822-830.

Brakebill, J. W., and Preston, S. E.. 1999. Digital data used to relate nutrient inputs to water quality in the Chesapeake Bay Watershed, Version 1.0. U.S. Geological Survey Water Open-File Report 99-60 U.S. Geological Survey, Baltimore, MD. <http://md.water.usgs.gov/publications/ofr-99-60> (April 21, 2009)

Brakebill, J. W., and Preston, S. D., 2003. A hydrologic network supporting spatially referenced regression modeling in the Chesapeake Bay Watershed. Environmental Monitoring and Assessment 81(1-3) 73-84.

Eck, D. J.. 2006. Introduction to Programming using Java, 5th ed. <http://math.hws.edu/javanotes> (April 21, 2009).

Ellson, J., Gansner, E., Koutsofios, L., North, S. C., and Woodhull, G., 2002. Graphviz - Open source graph drawing tools. Graph Drawing, 2265 483-484.

Hagberg, A. A., Schult, D. A., and Swart, P. J., 2008. Exploring network structure, dynamics, and function using NetworkX, in Proceedings of the 7th Python in Science Conference (SciPy2008), Gäel Varoquaux, Travis Vaught, and Jarrod Millman (Eds) Pasadena, CA USA, 11—15.

Johnes, P. J., 1996. Evaluation and management of the impact of land use change on the nitrogen and phosphorus load delivered to surface waters: The export coefficient modelling approach. Journal of Hydrology 183(3-4) 323-349.

Maidment, D. R. (Ed.), 2002. ArcHydro: GIS for Water Resources, ESRI Press, Redlands, Calif.

McMahon, G., Alexander, R. B., and Qian, S., 2003. Support of total maximum daily load programs using spatially referenced regression models. Journal of Water Resources Planning and Management-ASC*E*, 129(4), 315-329.

Moore, R. B., Johnston, C. M., Robinson, K. W., and Deacon, J. R., 2004. Estimation of total nitrogen and phosphorus in New England streams using spatially referenced regression models. U.S. Geological Survey Scientific Investigations Report 2004-5012, U.S. Geological Survey, Reston, VA. < http://pubs.usgs.gov/sir/2004/5012> (April 21, 2009)

Nolan, J.V., Brakebill, J. W., Alexander, R. B., and Schwarz, G. E., 2003. ERF1_2 – Enhanced River Reach File 2.0. U.S. Geological Survey Open-File Report 02-40. U.S. Geological Survey, Reston, Virginia. <http://water.usgs.gov/lookup/getspatial?erf1_2> (April 21, 2009)

Oliphant, T. E., 2007. Python for scientific computing. Computing in Science & Engineering, 9(3) 10-20.

Preston, S.D., and Brakebill, J.W., 1999. Application of Spatially Referenced Regression Modeling for the Evaluation of Total Nitrogen Loading in the Chesapeake Bay Watershed. U.S. Geological Survey Water Resources Investigations Report 99-4054, Baltimore, Maryland. < http://md.water.usgs.gov/publications/wrir-99-4054> (April 21, 2009)

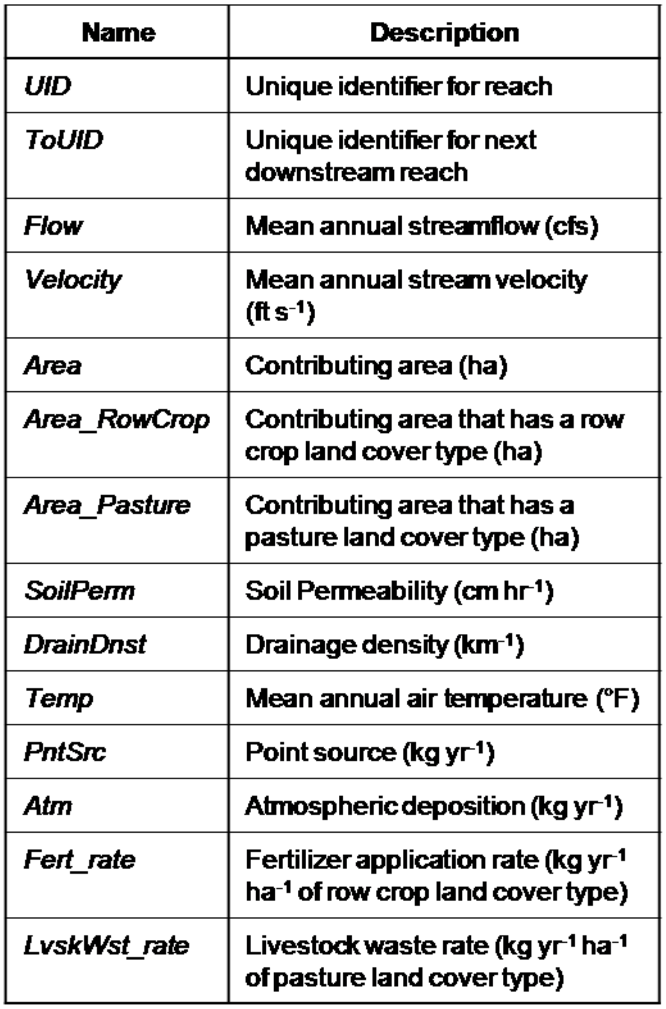
Robertson, D. M., Schwarz, G. E., Saad, D. A., and Alexander, R. B., 2009. Incorporating Uncertainty Into the Ranking of SPARROW Model Nutrient Yields From Mississippi/Atchafalaya River Basin Watersheds. Journal of the American Water Resources Association, 45(2) 534-549.

Schwarz, G.E., 2008. A Preliminary SPARROW model of suspended sediment for the conterminous United States. U.S. Geological Survey Open-File Report 2008–1205, U.S. Geological Survey, Reston, VA. < http://pubs.usgs.gov/of/2008/1205> (April 21, 2009)

Schwarz, G. E., Hoos, A. B., Alexander, R. B., and Smith, R. A., 2006. The SPARROW Surface Water-Quality Model: Theory, Application and User Documentation. U.S. Geological Survey Techniques and Methods, Book 6, Section B, Chapter 3, U.S. Geological Survey, Reston, VA.

Smith, R. A., Schwarz, G. E., and Alexander, R. B., 1997. Regional interpretation of water-quality monitoring data. Water Resources Research, 33(12) 2781-2798.

Table 1. Field names and descriptions of a pySPARROW database. These are the attributes required for applying the prediction portion of a SPARROW model.



Name	Description
UID	Unique identifier for reach
ToUID	Unique identifier for next downstream reach
Flow	Mean annual streamflow (cfs)
Velocity	Mean annual stream velocity (ft s^{-1})
Area	Contributing area (ha)
Area_RowCrop	Contributing area that has a row crop land cover type (ha)
Area_Pasture	Contributing area that has a pasture land cover type (ha)
SoilPerm	Soil Permeability (cm hr^{-1})
DrainDnst	Drainage density (km^{-1})
Temp	Mean annual air temperature (°F)
PntSrc	Point source (kg yr^{-1})
Atm	Atmospheric deposition (kg yr^{-1})
Fert_rate	Fertilizer application rate (kg yr^{-1} ha^{-1} of row crop land cover type)
LvskWst_rate	Livestock waste rate (kg yr^{-1} ha^{-1} of pasture land cover type)

Figure 1. SPARROW includes two primary transport mechanisms: (1) stream-to-water transport and (2) in-stream transport. Both are commonly modeled as nonlinear, first–order decay rates, although the mathematical structure of these transfer functions can be defined by the modeler.

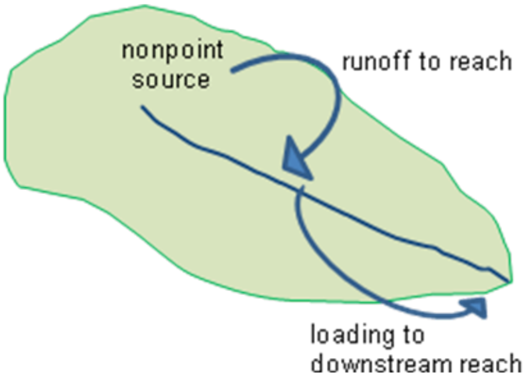


Figure 2. A Unified Modeling Language (UML) diagram for pySPARROW highlighting the Application Programmers Interface (API) classes and relationships between classes.

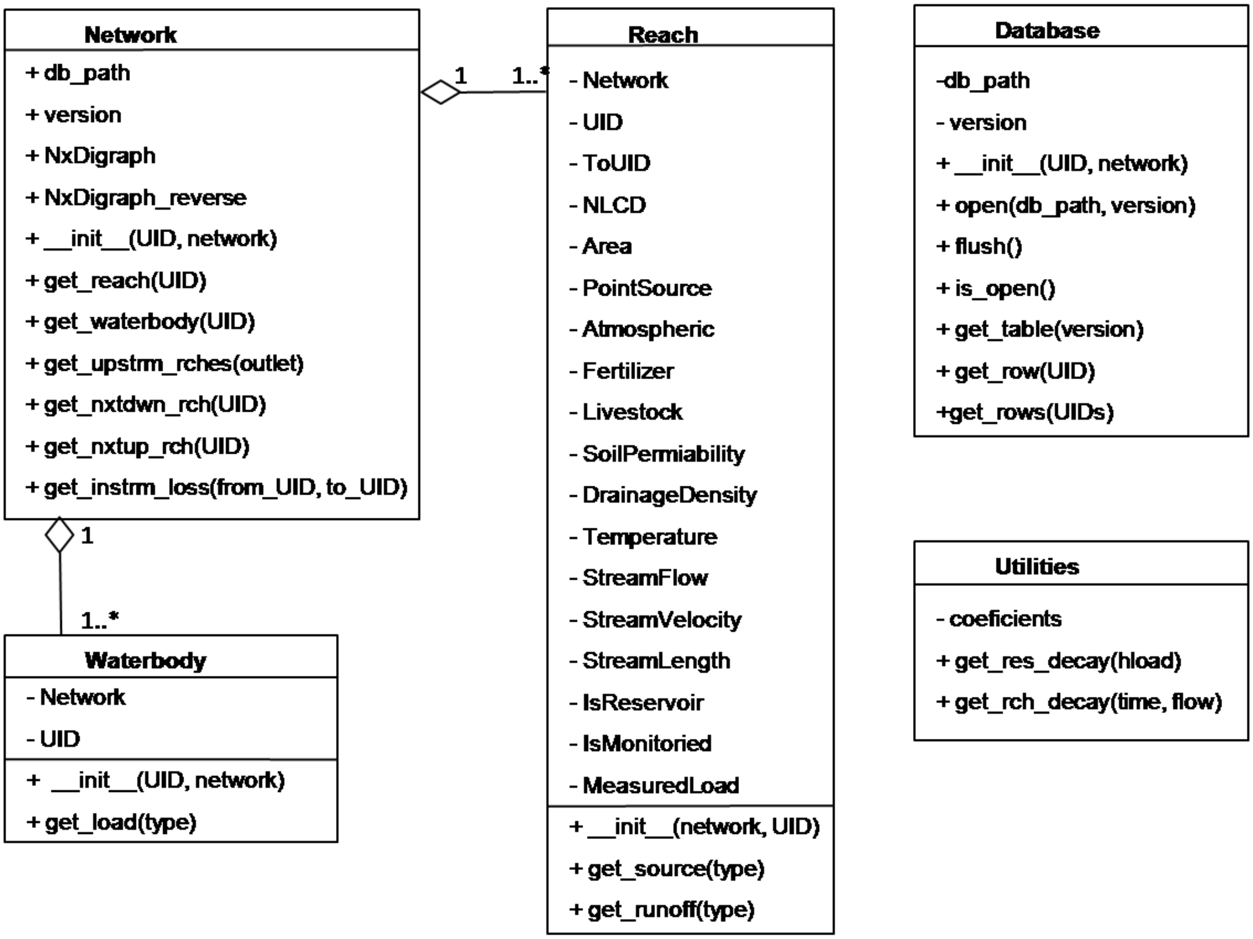


Figure 3. Upper Neuse River Basin, North Carolina in geographic space (a) and draw according to the GraphViz *neato* algorithm (b) to highlight tributaries flowing into the main branch of Neuse River. Part (b) of this figure is the structure used in forthcoming figures highlighting model results.

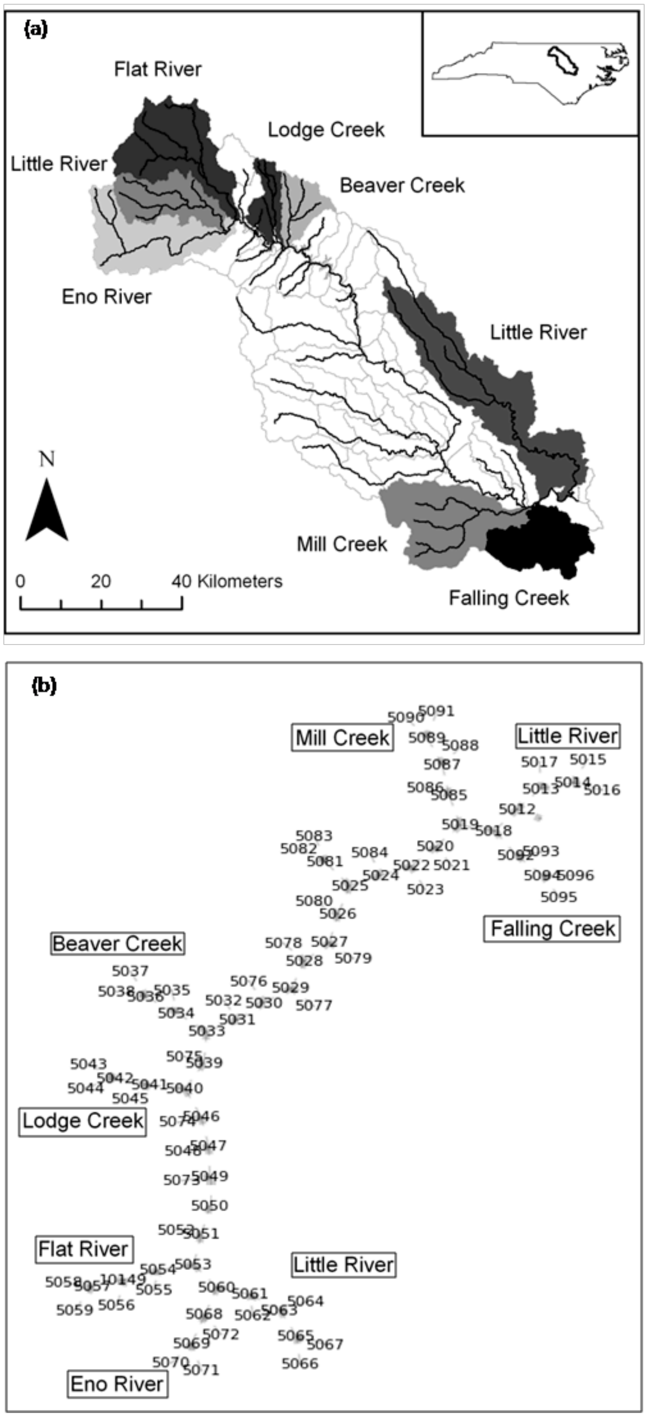


Figure 4. Results from Management Question 1 demonstrating the land-to-water transport process in pySPARROW for (a) total runoff and (b) source-specific runoff. Nodes represent river reaches with the node color representing to the incremental nitrogen [kg yr^{-1} ha^{-1}] delivered from the watershed to the stream within each reach. Note that node size is not scaled linearly to loading rate, but is drawn to instead emphasize differences between source-specific loading rates.

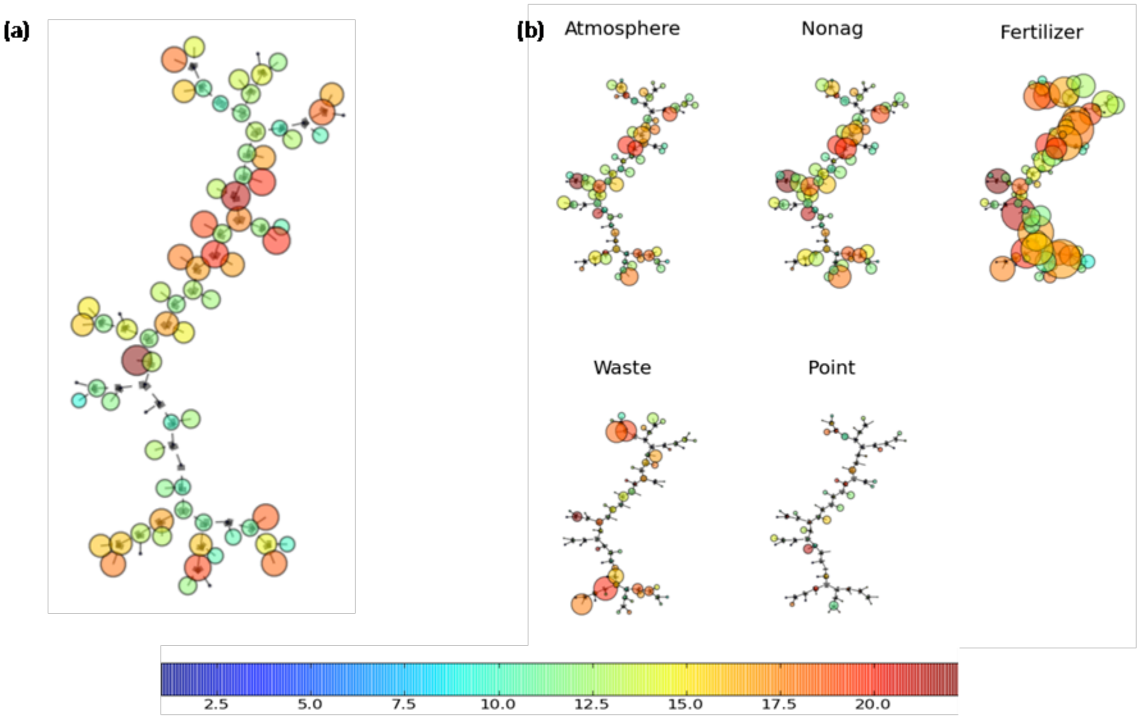


Figure 5. Results from Management Question 2 demonstrating the in-stream transport process in pySPARROW. The node colors and sizes represent a load reduction due to transport from that node to the basin outlet.

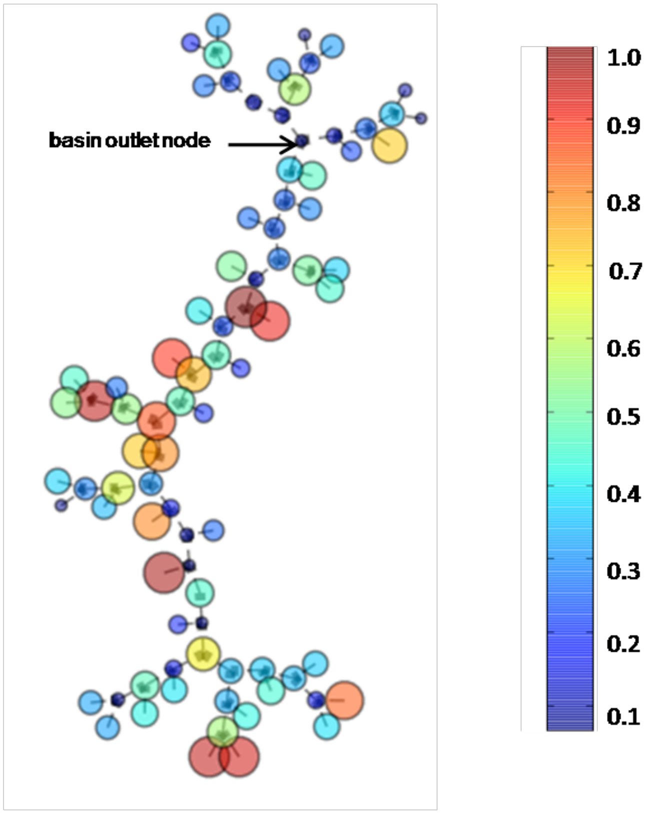
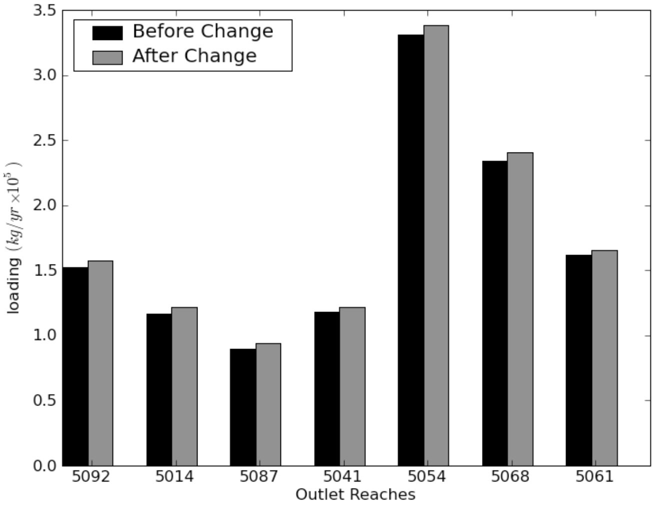


Figure 5. Results from Management Question 3 demonstrating the loading rate at the basin outlet before and after a 10% increase in fertilizer loading rates.


Before Change
After Change
loading $(kg/yr \times 10^5)$
Outlet Reaches