Design and Implementation of a General Software Library for using NSGA-II with SWAT for Multi-Objective Model Calibration

Mehmet B. Ercan
Water Resources Engineer, Arcadis, Indianapolis, IN
Previously
Research Assistant, Department of Civil and Environmental Engineering, University of South Carolina, Columbia, SC, USA.

Jonathan L. Goodall, Associate Professor, Department of Civil and Environmental Engineering, University of Virginia, Charlottesville, VA, USA.
And
Adjunct Professor, Department of Civil and Environmental Engineering, University of South Carolina, Columbia, SC, USA.

Corresponding Author: Jonathan L. Goodall, goodall@virginia.edu, PO Box 400742, Charlottesville, Virginia 22904, (434) 243-5019

Highlights

- We present an open-source software library for calibration of SWAT models
- The library implements the NSGA-II multi-objective genetic algorithm
- The library is used to calibrate a SWAT model of the Upper Neuse Watershed, NC
- The library can be used within SWAT-CUP for data visualization

Abstract

Calibrating watershed-scale hydrologic models remains a critical but challenging step in the modeling process. The Soil and Water Assessment Tool (SWAT) is one example of a widely used watershed-scale hydrologic model that requires calibration. The calibration algorithms currently available to SWAT modelers through freely available and open source software, however, are limited and do not include many multi-objective genetic algorithms (MOGAs). The Non-Dominated Sorting Genetic Algorithm II (NSGA-II) has been shown to be an effective and efficient MOGA calibration algorithm for a wide variety of applications including for SWAT model calibration. Therefore, the objective of this study was to create an open source software library for multi-objective calibration of SWAT models using NSGA-II. The design and implementation of the library are presented, followed by a demonstration of the library through a test case for the Upper Neuse Watershed in North Carolina, USA using six objective functions in the model calibration.

Keywords: Multi-Objective Calibration; Genetic Algorithms; Watershed Modeling; SWAT; NSGA-II

Software availability: The software is available free and open source on Github: *https://github.com/mehmetbercan/NSGA-II_Python_for_SWAT_model*.

1. Introduction

The Soil and Water Assessment Tool (SWAT) is a widely used watershed model with numerous applications around the world for water quantity and quality simulations (e.g., Cools et al., 2011; Gassman et al., 2007; Liu et al., 2013). It can be classified as a semi-distributed conceptual watershed model that is capable of running on a daily or sub-daily time step over long time periods. SWAT is able to simulate large watersheds with different management scenarios where the impact on water supply and non-point source pollution can be assessed (Arnold et al., 1998). For SWAT and other similar watershed models, there are often hundreds of modeling units in a model for a single watershed and dozens of model parameters used to describe properties within the model. One of the modeler's most important and difficult tasks is to calibrate these model parameters so that the model's output matches observational data such as streamflow observations collected within the watershed.

Many algorithms and tools have been developed and applied for calibrating SWAT models. SWAT-CUP represents one widely used tool in the SWAT community for applying calibration algorithms to SWAT models. SWAT-CUP includes different calibration algorithms, as well as routines for sensitivity analysis, validation, and uncertainty analysis of SWAT models (Abbaspour et al., 2007). There are other procedures and algorithms developed in the scientific community for calibration that have not yet been included in SWAT-CUP, but that would benefit SWAT modelers. For example, SWAT-CUP does not include multi-objective calibration approaches, nor does it include genetic algorithm calibration approaches (Abbaspour, 2013). SWAT modelers, however, could benefit from these calibration procedures, especially for large watersheds where multiple streamflow observations are available (Arnold et al., 1999; Bekele and Nicklow, 2007; Kirsch et al., 2002; Santhi et al., 2001; White and Chaubey, 2005).

Genetic Algorithms (GAs) offer the ability to effectively solve highly non-linear optimization problems and have been used for a variety of water resources challenges. Being an evolutionary algorithm, GAs use principles of genetics and natural selection for optimization (Haupt and Haupt, 2004). They are well suited for hydrologic models, which usually cannot be adequately calibrated by gradient-based calibration algorithms. The objective function for each solution in a GA can be evaluated in parallel computations, which provide computational advantages (Zhang et al., 2013, 2012a). The heuristic search procedure of GAs, relying on stochastic search rules, increases the probability of finding non-unique solutions. Previous studies have shown that these properties of GAs allow them to converge to optimal solutions for a variety of problems (Winston et al., 2003) including the challenge of calibrating watershed-scale hydrologic models (Arabi et al., 2006; Nicklow and Muleta, 2001).

Multi-objective calibration algorithms have been shown to increase model performance for hydrologic models of large watersheds (Andersen et al., 2001). In contrast to the more widely used single-objective calibration algorithms available to SWAT users now in tools like SWAT-CUP, multiple-objective calibration better constrains the calibration process, resulting in a calibrated model that better matches the physical conditions within the watershed (Niraula et al., 2012). Watershed models may use multiple objective functions in a calibration procedure to account for potentially competing objectives, even for cases when only a single streamflow station is available for calibration (e.g., two objectives might be to match peak flows and maintain annual water volume balance between the model and observations). They can also allow modelers to take advantage of multiple observational time series (e.g., streamflow at two or more locations in the watershed or streamflow and soil moisture observations at two or more locations in the watershed).

There is a class of calibration routines that combine the benefits of both multi-objective and genetic algorithm calibration approaches: the so called multi-objective genetic algorithms (MOGAs). One of the most popular MOGAs is the Non-Dominated Sorting Genetic Algorithm II (NSGA-II). NSGA-II is a fast and efficient population-based optimization technique that can be parallelized. The algorithm has been shown to be superior to other MOGAs (Deb et al., 2002; Zitzler et al., 2000) and it has the potential to reduce calibration time through efficiency in the algorithm itself and its ability to easily be mapped to parallel computing resources (Deb et al., 2002; Tang et al., 2006; Zitzler et al., 2000). The algorithm has significant improvements over the original NSGA (Srinivas and Deb, 1994) including adding elitism, reducing the complexity of the non-dominated sorting procedure, and replacing a sharing function with a crowded-comparison function. The NSGA-II algorithm has also been shown to be an effective tool for watershed model calibration (Bekele and Nicklow, 2007; Confesor and Whittaker, 2007; Hejazi et al., 2008; Kayastha et al., 2011; Khu and Madsen, 2005; Lu et al., 2014; Shafii and Smedt, 2009; Zhang et al., 2012b).

While NSGA-II has been used for calibrating watershed models, there is no known software implementation of NSGA-II for calibrating SWAT models that is freely available to the community. One study did report creating a multi-objective calibration tool for SWAT models using NSGA-II (Bekele and Nicklow, 2007). However, based on personal communication with the authors, the source code for this implementation is no longer available. The goal of this work, therefore, is to create an open source and freely-available NSGA-II software library for SWAT model calibration. We designed the tool to be library that can be used alone or incorporated into other software tools. We specifically designed the software to be easily integrated into SWAT-CUP given the popularity of this tool with the SWAT community. We chose to implement the library using the Python programming language because of its growing popularity in the scientific computing community.

In the remaining sections of this paper, we first describe the algorithm for using NSGA-II with SWAT for model calibration, then describe the design and implementation of the NSGA-II/SWAT library including compatibility with SWAT-CUP, and finally present a test case application of the library for calibrating a SWAT model of the Upper Neuse watershed in North Carolina. As part of this test case application, we compare the results of the NSGA-II calibration to results from a single-objective calibration to show the improvement obtained by using the multi-objective NSGA-II algorithm. We have provided the source code for the NSGA-II/SWAT library as an open source and freely available repository through GitHub: *https://github.com/mehmetbercan/NSGA-II_Python_for_SWAT_model*.

2. The NSGA-II Algorithm and its Integration with SWAT

2.1 Overall Process Flow

In this section we explain the NSGA-II algorithm and how we integrated SWAT calibration into the algorithm when designing the NSGA-II/SWAT library. Our approach follows the example of past work using NSGA-II for SWAT calibration (e.g., Bekele and Nicklow, 2007; Kayastha et al., 2011; Lu et al., 2014), but extends this past work to create a general and reusable software tool. For further detail on the NSGA-II algorithm itself, readers are referred to Deb et al. (2002). For convenience, we provide a mapping between NSGA-II and SWAT calibration terminology in Table 1.

Table 1: Description of NSGA-II terms as they relate to SWAT calibration

NSGA-II Term	Description for Application to SWAT Calibration
Solution	An individual of a population that includes a SWAT calibration parameter set and NSGA-II processing data for the parameter set
Gene	The SWAT calibration parameter set that exists in a solution
Chromosome	An individual of a gene that represents a single SWAT calibration parameter
Binary Value	Binary representation of chromosome in a user defined number of bits

A standard NSGA-II process typically begins with a random parent population P_i (Deb et al., 2002). However, here we start with a Latin Hypercube Sampling (LHS) (See Step 1 in Figure 1) because better results have been achieved for SWAT models using this approach (Bekele and Nicklow, 2007). The LHS operator is executed first to create an initial combined population ($R_{i=0}$). We use the subscription "i" to represent a generation (iteration) number. The initial combined population must be at least twice as large the population size for reasons that will become clearer in forthcoming steps of the algorithm.

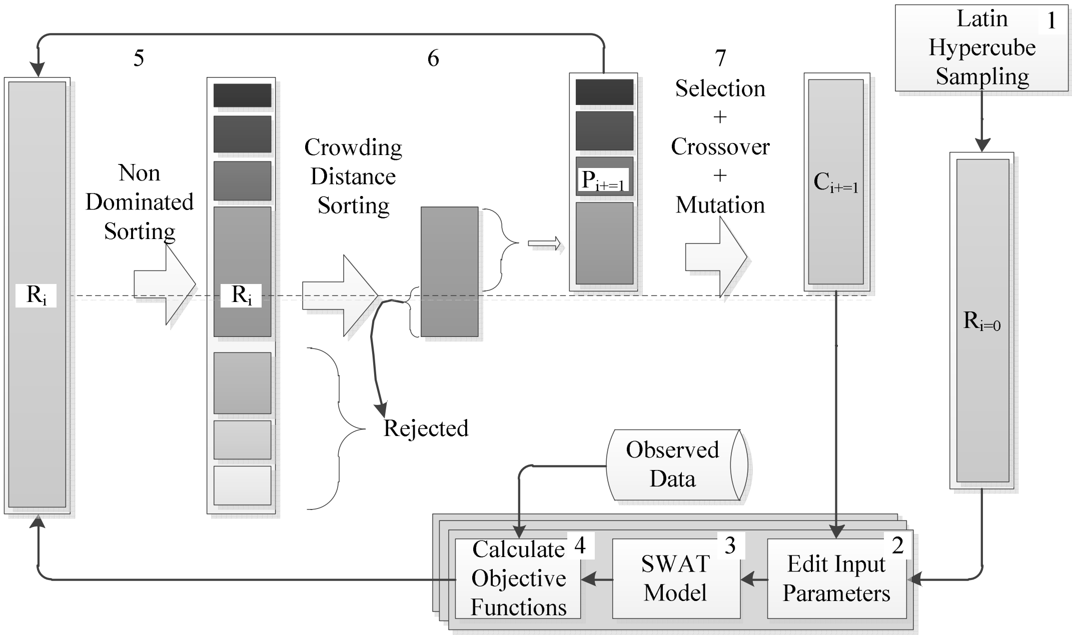

Figure 1: The NSGA-II algorithm for SWAT model calibration.

Each solution in the initial combined population ($R_{i=0}$) is considered to be a SWAT calibration parameter set. The SWAT input files are edited to include this solution, the model is executed, and the objective functions are evaluated using observational data and the SWAT model output data (See Steps 2-4 in Figure 1). These model runs can be performed in parallel for each solution within the population. Once this process has been completed, the solutions within the population (R_i) are ranked using the results of the objective function evaluation process and a non-dominating sorting approach (See Step 5 in Figure 1). Details of this non-dominating sorting approach are provided in Section 2.2.1.

The best performing solutions from R_i as determined by the non-dominating sorting approach are used to form the parent population (P_i). The number of solutions in the parent population is determined by the user defined population size. In the case of ties where multiple solutions exist with the exact same ranking at the cut-off point for creating P_i, a crowded distance sorting operator is used to break the tie (See Step 6 in Figure 1). This operator is explained in Section 2.2.2. In short, the solutions with the larger crowding distance value, which acts as a dummy fitness in the sorting operator, are chosen to fill the remaining spots in P_i. Using the parent population, a new child population ($C_{i+=1}$) is determined through a selection, crossover and mutation operator (See Step 7 in Figure 1), which is explained in Section 2.2.3. This entire procedure is repeated until the termination criteria are met.

2.2 NSGA-II Operators

We provide in this section details for the specific operators used in the NSGA-II algorithm that are mentioned in the previous section.

2.2.1 Non-Dominated Sorting

The non-dominated sorting operator is a process of ranking solutions that exist in the combined population (R_i) (Deb et al., 2002; Srinivas and Deb, 1994). In this operator, the objective functions are evaluated for given solutions to determine domination. Domination is established when the objective function evaluations of a solution outperform all other solutions with the same rank. The process terminates when all members of the combined population (R_i) have been assigned a rank.

2.2.2 Crowding Distance Sorting

Crowding distance sorting is used to break ties for solutions with the same rank at the cut off point for being included in the parent population (P_i) (Deb et al., 2002). First, the solutions in that rank are sorted based on the value of an objective function. Then, a solution is selected and the distance between that solution and each of the adjacent solutions is calculated. These distances are normalized by dividing by the distance between the maximum and minimum value of the objective function for all solutions. Finally, crowding distance for the solution is calculated as the sum of the normalized distance for the adjacent solutions.

This process is repeated for all objective functions and the final crowding distance value for a solution is the summation of crowding distances calculated for all objective functions. It is then repeated for all solutions within the parent population. One exception is the maximum and minimum solutions in a rank. Because they do not have adjacent solutions on both sides, they are typically assigned an arbitrarily large distance value. When breaking ties, the preference is to select solutions with a large crowding distance value, which means the solution has more distant neighbors and selecting this solution helps to protect the diversity of the population.

2.2.3 Selection, Crossover, and Mutation

Selection is a process that chooses solutions from a parent population ($P_{i+=1}$) that go into a child population ($C_{i+=1}$) based on non-dominated and crowding distance sorting values. It starts by randomly selecting two solutions from $P_{i+=1}$. Then, it selects the solution that has the smaller rank. If two solutions have the same rank from non-dominated sorting, it selects the solution that has the greater crowding distance value. This process continues until all spots in $C_{i+=1}$ are filled.

After completion of the selection process, the crossover process begins. There are two techniques for the crossover operation: regular crossover and uniform crossover. In regular crossover, each pair of adjacent solutions from $C_{i+=1}$ are progressively chosen. Then, a random number is generated and compared to a crossover probability. If the random number is smaller than the crossover probability, crossover occurs where chromosomes between the two solutions flip for a randomly generated number of chromosomes.

Uniform crossover is different from regular crossover in that the crossover happens at a binary level instead of at a solution level. The uniform crossover goes through all binary values (0 or 1) (of chromosomes) for every evenly indexed $C_{i+=1}$ solution. Uniform crossover happens if a random number is smaller than the crossover probability. In this case, the binary value is replaced with the binary value from the corresponding next (oddly indexed) $C_{i+=1}$ solution.

Finally, mutation happens through $C_{i+=1}$ solutions at a binary level similar to uniform crossover. The mutation process simply flips the binary value (from 1 to 0, or vice versa) if a random number is smaller than the mutation probability.

3. Design and Implementation of the NSGA-II/SWAT Calibration Library

The NSGA-II/SWAT calibration library implements the algorithm summarized in the prior section where NSGA-II was used for SWAT model calibration. The library was designed as a general, object-oriented application programming interface (API) library and implemented in the Python programing language because it is open source and widely used in scientific communities. The library was tested against an established NSGA-II implementation written in the C programing language (Deb et al., 2002) to ensure that it is able to reproduce the same results. The library was designed to be compatible with SWAT-CUP (Abbaspour, 2013; Abbaspour et al., 2007), which is a widely used tool for calibration of SWAT models, as described later in this section.

3.1 Class Diagram

The NSGA-II/SWAT calibration library includes one main class called nsga2 and two utility classes for lower level NSGA-II and SWAT operations (Figure 2). The nsga2 class is heart of NSGA-II algorithm and includes operations such as creating child and parent populations.

During the initialization phase, the nsga2 class stores inputs such as population size, genes, chromosomes, and objective functions provided by the user. The nsga2 class offers two options for creating an initial combined population ($R_{i=0}$): (i) using the Latin Hypercube Sampling (LHS) method and (ii) reading the last generation from a previous calibration. The LHS method is included because, as stated earlier, it creates a better initial solutions for SWAT models (Bekele and Nicklow, 2007). On the other hand, reading the last generation from the previous calibration allows users to continue from previous but ultimately unsuccessful calibrations (for example, if a calibration fails to complete midway through the calibration process).

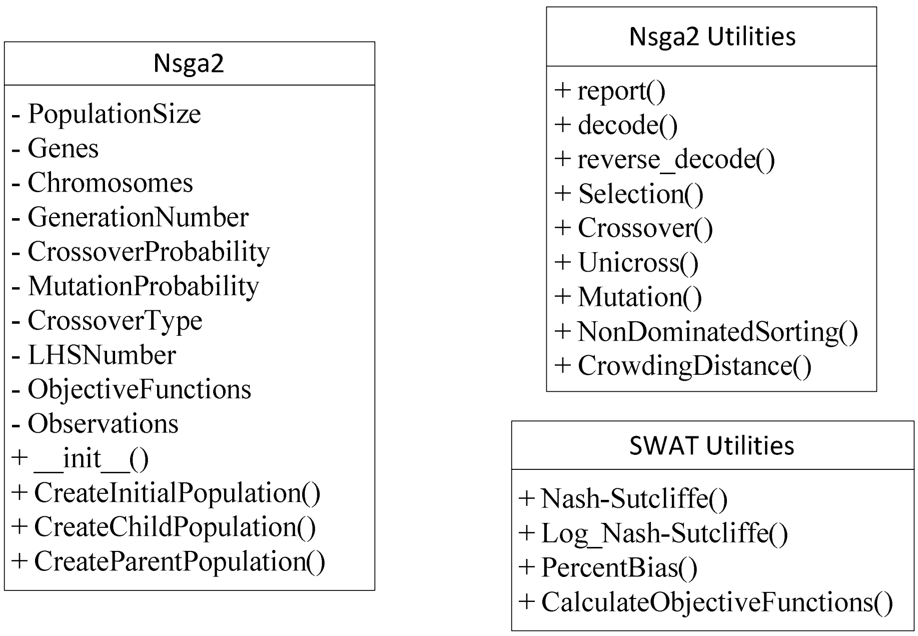

Figure 2: The NSGA-II/SWAT calibration library design.

The utility classes supplement the calibration process by providing lower-level functionality specific to the NSGA-II algorithm and for communication with SWAT. The nsga2 class uses nsga2 utilities to complete methods such as *Crossover()* or *Unicross()* required when creating child populations based on the user's choice along with *Selection()* and *Mutation()* methods.

Similarly, creating a parent population requires methods like *NonDominatedSorting()* and *CrowdingDistance()*, which are also implemented in the nsga2 utility class. SWAT utilities are used for objective function calculations using methods like *Nash-Sutcliffe()* and *PercentBias()*. By separating the SWAT-specific functionality into its own class, our design goal was to provide a pattern that could be repeated when expanding the library to support other hydrologic models.

3.2 Application for SWAT Calibration

To obtain SWAT model parameter values (genes), the binary values of chromosomes from solutions of C_i go through a decoding process (*decode()*). Then, the SWAT model input files are ready to be edited and executed to calculate objective functions using the SWAT utility class method, *CalculateObjectiveFunctions()*. This method first creates a *model.in* file containing genes. Then, it executes a batch file called *nsga2_mid.cmd* that creates the *model.out* file by using the *model.in* file and the SWAT model engine. Finally, the *CalculateObjectiveFunctions()* method uses the *model.out* file and calculates the objective function values by using other SWAT utility functions such as *Nash-Sutcliffe()*. This process continues until each solution of C_i is assigned objective function values.

The *nsga2_mid.cmd* file is a batch file that executes a series of commands for SWAT calibration. It uses SWAT executable (*swat.exe*) and two Python scripts (*SWAT_ParameterEdit.py* and *Extract_rch.py*) in order to create the *model.out* file. It first runs *SWAT_ParameterEdit.py* to change SWAT model parameters based on information in *model.in* file. Then, it executes *swat.exe* to execute the SWAT model using the parameter values included in the *model.in* file. Finally, it runs *Extract_rch.py* to extract SWAT model outputs into *model.out* file. The *nsga2_mid.cmd* file gives flexibility to edit the SWAT side of the calibration procedure. To illustrate, inorganic nitrogen flux is the sum of nitrite (NO2) and nitrate (NO3), which SWAT prints separately. Thus, an intermediate script could be inserted in *nsga2_mid.cmd* file to sum these two nitrogen flux terms in *model.out* file for use in later calibration steps.

3.3 Compatibility with SWAT-CUP

The NSGA-II/SWAT calibration library was designed so that it can be integrated into SWAT-CUP. First, we included a *Backup* folder as a reference to default parameter values as done in SWAT-CUP. The input/output file and folder names were created following the SWAT-CUP pattern. For example, the *SWATtxtInOut* folder contains the NSGA-II input and output folders named *NSGA2.IN* and *NSGA2.OUT*. We further followed SWAT-CUP patterns by creating files with the same structure. The calibration parameter definition file (*nsga2_par.def*) is named with the calibration method and followed with *_par.def*. The structure of *nsga2_par.def* file is defined as "*X__parameter.ext min max*" where the *X* defines the parameter editing method, the *parameter* defines the SWAT parameter, the *ext* defines the extension of SWAT files, and the *min* and *max* define the minimum and the maximum parameter limits.

In addition to the structure and naming conventions, internal parts of the NSGA-II/SWAT library also follow the SWAT-CUP pattern. The *SWAT_ParameterEdit.py* script is equivalent to *SWAT_edit.exe* of SWAT-CUP. Both scripts edit SWAT files based on the *model.in* file created by the calibration algorithm. Also, the *Extract_rch.py* script is equivalent to SWAT-CUP's extracting script, *Extract_rch.exe*, which extracts SWAT outputs into *model.out* file in the equivalent format. The batch file (*nsga2_mid.cmd*) mentioned in a prior section (which also exists in SWAT-CUP) can be used to run extensive SWAT-CUP editing and extracting executable files, rather than our parameter editing and extracting scripts. All these properties were intentionally included to ease the integration of our software library into SWAT-CUP.

Instructions for running NSGA-II through SWAT-CUP are provided on the software's GitHub site. The basic procedure is to override the GLUE method and replace it with the NSGA-II method. This is not a long-term solution, but rather a proof-of-concept solution that does not require altering the SWAT-CUP code-base. Later work can easily extend this proof-of-concept by allowing SWAT-CUP to include both the NSGA-II method along side the existing methods. The output generated by NSGA-II conforms to a structure expected by SWAT-CUP, allowing users to visualize the calibration results like any other calibration routine currently within SWAT-CUP (Figure 3).

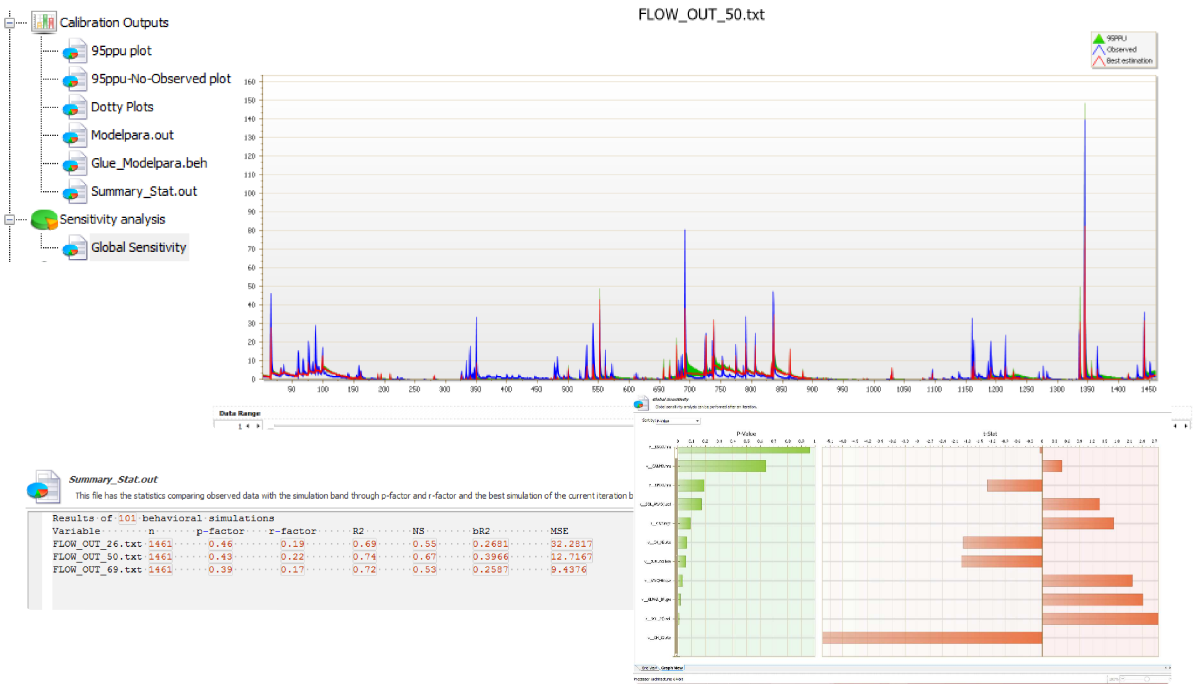

Figure 3: Example visualizations of NSGA-II calibration results through SWAT-CUP Graphical User Interface (GUI).

This proof-of-concept could be formalized by modifying SWAT-CUP so that new calibration routines can be plugged-in without the need to recompile the core SWAT-CUP code. This plug-in architecture would allow third-party developers to create calibration routines to be added to the software system more easily. Given the existing capabilities of SWAT-CUP for data management and visualization, a plug-in architecture could be very powerful for incorporating the latest calibration methods and providing them to SWAT modelers in a convenient and familiar. It would save the work of recoding the visualization capabilities already available through SWAT-CUP and provide a consistent UI experience for end users. Libraries like the NSGA-II/SWAT library created in this study could be easily structured to follow a standard required for integrated into SWAT-CUP as a plug-in.

4. Test Case

The NSGA-II/SWAT library is demonstrated for a test case application using a SWAT model of the Upper Neuse Watershed in North Carolina. The library is used to calibrate this model to match streamflow records at three observation sites using two fitness criteria. In the following subsections, we first briefly discuss how we created a SWAT model for Upper Neuse watershed, second show how we used our NSGA II library to calibrate the SWAT model, and third present the results of the calibration. The primary goal of this section is to illustrate how the library would work for end users interested in applying the library to calibrate a SWAT model. A secondary goal is to explore how the model calibration resulting from using the NSGA-II/SWAT library compares to the widely used single-objective calibration strategy.

4.1 Study Area and Model Preparation

The Upper Neuse watershed (Figure 4) is a level-8 watershed that includes the Flat, Little, and Eno River watersheds defined by the United States Geological Survey (USGS) codes 02085500, 0208521324 and 02085070, respectively. The study area has a mild climate and gently rolling topography. The soil type of the watershed is dominated by silty clay and loam, and the land cover of the watershed is dominated by forest and cultivated crops.

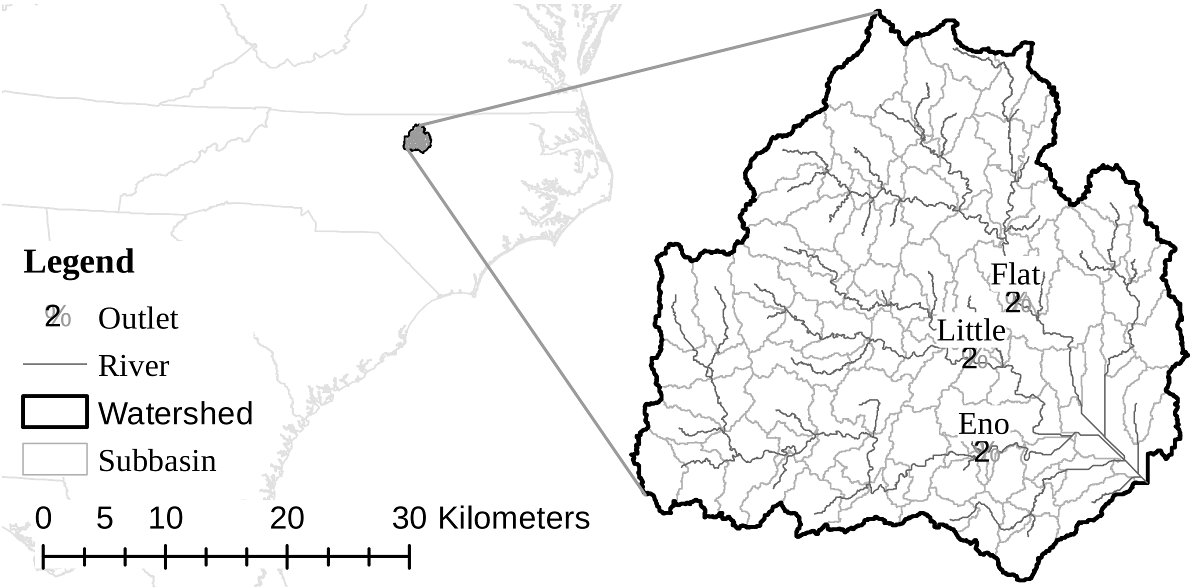

Figure 4: Study area: the Upper Neuse Watershed in North Carolina, USA.

Terrain and land cover data were obtained from the United States Geological Survey (USGS) National Elevation Dataset (NED) and the 2006 version of the National Land Cover Database (NLCD). Soil data were obtained from the State Soil Geographic (STATSGO) dataset provided by the United States Department of Agriculture (USDA). Air temperature, wind speed, and humidity were obtained from the National Climatic Data Center (NCDC). Precipitation data was obtained from National Weather Service (NWS) for Nexrad-derived rainfall estimates and from NCDC for gauge observed rainfall estimates. These two precipitation estimates were combined using the approach described by Ercan and Goodall (2012) to create a composite rainfall dataset for the watershed area. Lastly, daily average streamflow data from the USGS National Water Information System (NWIS) were downloaded using the Consortium of Universities for the Advancement of Hydrologic Science, Inc. (CUAHSI) Hydrologic Information System (HIS) (Tarboton et al., 2009).

We divided the watershed into subbasins based on the USGS streamflow station locations and homogeneity of land characteristics. We used threshold values of 10% for soil, slope, and land cover to reduce variability within the subbasins. The result was a total of 837 Hydrologic Response Units (HRUs) for the 93 subbasins in the watershed, which is within the HRU/subbasin ratio range recommended in SWAT documentation. The commonly used settings were chosen to configure the model that include the Natural Resources Conservation Service (NRCS) Curve Number (CN) surface runoff method, the Penman-Monteith potential evapotranspiration method, and the variable storage channel routing method. The ArcSWAT software program was used for much of the data preprocessing steps required to create the model.

4.2 Model Calibration

Streamflow observations at the Flat, Little, and Eno watershed outlets were used in the calibration. For each outlet, the Nash-Sutcliffe (E) and Percent Bias (PB) statistics were used as measures of the goodness of fit. Therefore, the calibration used six objective functions (3 sites x 2 fitness). We ran Generalized Likelihood Uncertainty Estimation (GLUE) (Beven and Binley, 1992) available in SWAT-CUP to find the sensitivity of the flow parameters on streamflow prediction. The six most sensitive parameters were chosen for model calibration with the acceptable ranges and replacement operations shown in Table 2.

Table 2: Model parameters, their calibrated values, acceptable ranges, and replacement operations

Parameter	Value	Range	Operation
Alpha_Bf	0.99	0.01-1.00	Replaced
Cn2	0.07	±0.25	% Relative
Ch_K2	30.59	0.01-150.00	Replaced
Canmx	9.53	0.01-10.00	Replaced
Esco	0.94	0.01-1.00	Replaced
Sol_Aw c	-0.06	±0.25	% Relative

We used the following settings for calibrating the Upper Neuse watershed model with NSGA-II. The LHS size was set to 1000 and crossover probability was set to 0.5 using uniform crossover. The mutation probability and the seed for the random number generation were set to 0.5. Population size and generation number were set to 80. Since our parameters do not have a wide range, we used 8 bits for binary crossover and mutations.

Figure 5 provides the pseudo code for the NSGA-II calibration to briefly illustrate how it was used in the case study. The first line initializes the nsga2 class, which reads in the inputs from the *SWATtxtInOut* folder such as *PopulationSize*, *GenerationNumber* and *Observations*. Then the initial combined population is created followed by the generation loop. In the generation loop, the code first creates the parent population from the combined population. Second, it creates the child population using the parent population. Then the child population is used to run the SWAT model and the model's output is used to evaluate the objective functions.

Finally, the parent and child populations are used to create the new combined population for the next generation. As seen in Figure 4, this library can easily be adapted to other watershed simulation models by modifying the initialization method of the nsga2 class and the *CalculateObjectiveFunctions()* process that exists in the SWAT utility class.

```
NSGAII = Nsga2.nsga2(SWATtxtInOut)
R_i=0 = NSGAII.CreateInitialPopulation()
R_i=0 = SWATUtilities.CalculateObjectiveFunctions(R_i=0)
FOR i = 0 to NSGAII.GenerationNumber
    P_i = NSGAII.CreateParentPopulation(R_i)
    C_i = NSGAII.CreateChildPopulation(P_i)
    C_i = SWATUtilities.CalculateObjectiveFunctions(C_i)
    R_i+1 = P_i + C_i
END FOR
```

Figure 5: The pseudo code for applying the NSGAII/SWAT library for calibrating the test case SWAT model.

4.3 Calibration Results

The Pareto front solutions for the case study example are shown in Figure 5. There are six objective functions for 80 solutions. The objective functions are percent bias (PB) and one minus Nash-Sutcliffe (1-E) for the stations at the outlets of the Flat, Little and Eno watersheds. The number of solutions is defined by the population size because all solutions in the final generation are in the first front (ranking). A zero value on the figure indicates an optimal result while higher values indicate worse model efficiency. The figure shows the range in performance of the three watersheds in terms of PB and 1-E values. The values ranged between 0.00 and 0.39 for PB and between 0.23 and 0.88 for E across the three observation sites.

We highlighted the tradeoffs in Figure 6. The thick black line shows the solution selected with an equal weight for all objective functions, defining the best possible solutions considering all three objective functions equally. When we put a large weight on the 1-E objectives, we get the thick dashed grey line that slightly improves on 1-E values, but is worse for PB values. In the last case with the thick grey line, we selected the lowest 1-E value (best E) for the Eno watershed ignoring all other criteria. In this case, which represents calibration using a single objective function, the E value improves for the Eno watershed as expected, but the other objective functions, including PB for the Eno watershed, are worse compared to the equally weighted multi-objective case.

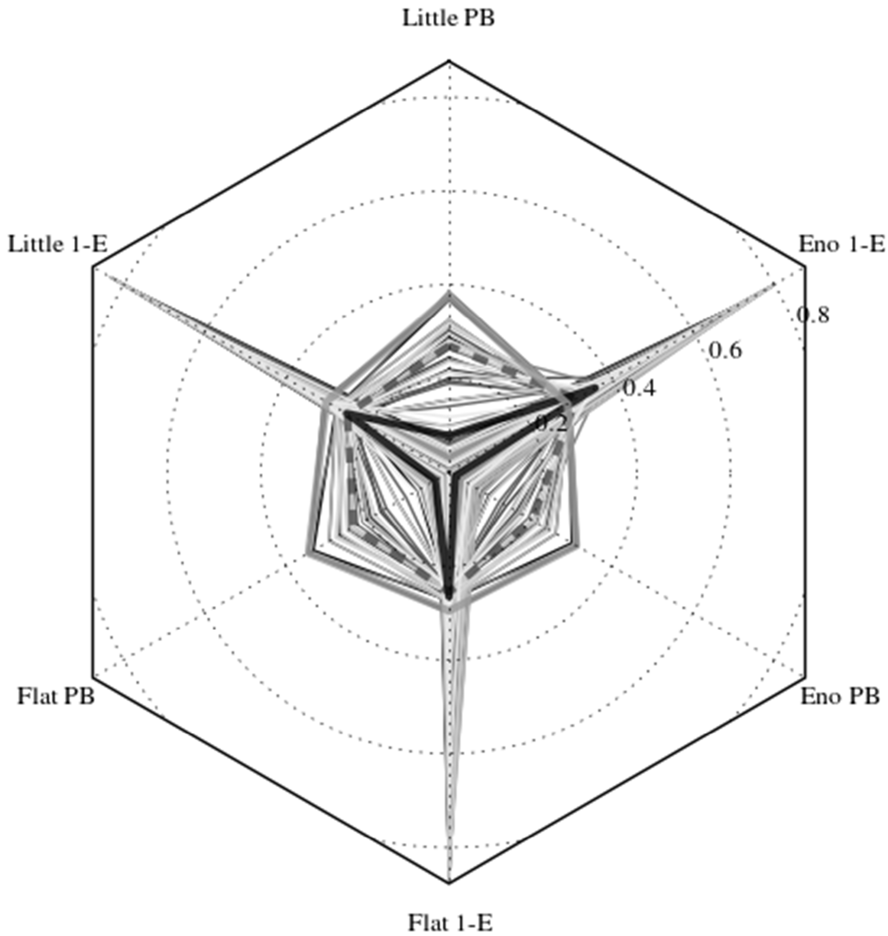

Figure 6: Six dimensional NSGA-II Pareto front.

For visualization of tradeoffs, we displayed the same Pareto front in Figure 6 using two dimensional graphs. Because of difficulties of showing all six objective functions on a single graph, we averaged fitness values over the Flat, Little and Eno watershed outlets in Figure 7b. Significant tradeoffs are illustrated between E and PB objective functions for the three outlets (Figure 7a) as was also shown by Bekele and Nicklow (2007). This illustrates the utility of a multi-objective calibration of SWAT models by attempting to balance multiple competing objectives when selecting optimal parameter sets.

The equally weighted objective functions are also highlighted in Figure 7. Better PB and 1-E values exist on Figure 7a. However, these values are connected to other objective functions that are much worse (e.g. the grey dashed and solid lines in Figure 6). Figure 7a indicates similar responses between the three watersheds, but a more significant relationship between the Flat and Little watersheds. This is expected as all the watersheds are in the same region and the Eno watershed is partially urbanized whereas the Flat and Little are not.

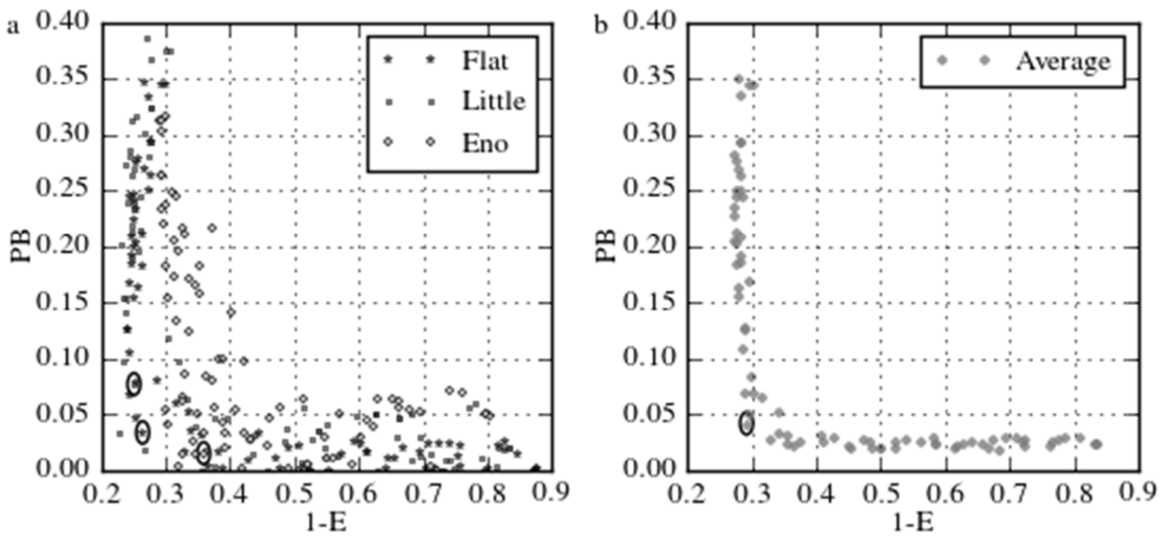

Figure 7: (a) NSGA-II Pareto front with (b) results averaged across the three watersheds.

Table 2 shows the parameter set values for the chosen solution (objective functions are equally weighted). We ran the SWAT model based on this solution and prepared the model statistics against observations (Table 3). The daily and monthly statistics showed good agreement between simulated and observed streamflows for each site. PB values are considered to be "very good" for both the calibration and validation periods except for the Flat River watershed during the validation period, which is considered to be "good" (Moriasi et al., 2007). Monthly E values, on the other hand, were considered to be "good" for the calibration period and "very good" for the validation period (Moriasi et al., 2007). Lastly, daily statistics showed very good accuracy compared to previous SWAT studies (Gassman et al., 2007), indicating the strength of the calibration method.

Table 3: Results of the fitness values during the calibration and evaluation time periods for the Flat, Little, and Eno watersheds.

	2005-2008[a]					2009-2012[b]				
Watershed	E	E^c	R^2	R^{2c}	PB	E	E^c	R^2	R^{2c}	PB
Flat	0.74	0.73	0.75	0.74	0.04	0.62	0.8	0.62	0.82	-0.13
Little	0.75	0.72	0.76	0.73	0.08	0.61	0.8	0.61	0.81	-0.09
Eno	0.65	0.65	0.73	0.7	0.02	0.59	0.77	0.64	0.82	-0.11

a Calibration period
b Evalutation period
c Daily predicted and observed values aggregated to monthly

The solution with the equally weighted objective functions within the Pareto front is also illustrated in Figure 8. Similar to Table 3, the Little and Flat watersheds are slightly better at matching high flows (better E value) compared to the Eno watershed. All of the watersheds tend to underestimate streamflow for the calibration period and overestimate streamflow for the evaluation period. In general, the monthly accumulated streamflow values support the accuracy of the model as both the calibration and evaluation periods generally fit well to observed streamflow for all three sites.

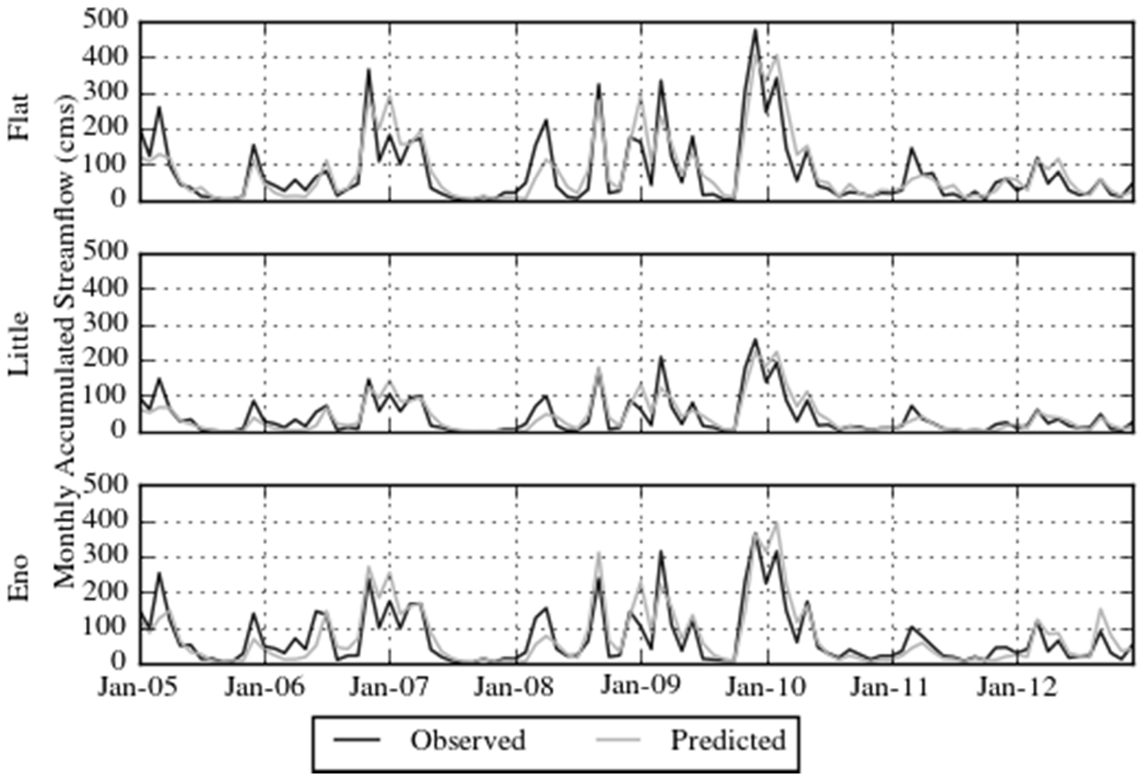

Figure 8: Comparison of monthly simulated and observed streamflow.

Finally, we examined the solution with the best E value for Eno watershed (highlighted with the thick grey line in Figure 6). This case is equivalent to single-objective calibration as we selected a solution with regard to only one objective function and ignored all other objective functions. When using this parameter set, the E value for the Eno watershed improved by 0.06 and 0.02 for calibration and validation periods, respectively, compared to the results when using the parameter set from the equally weighted multi-objective solution. However, all other statistics for the calibration and validation period for the three watersheds decreased when using the parameter set from the single objective optimization. The magnitude of decrease in fitness values was often similar to the gain in E for the Eno watershed. However, the PB values deteriorated into an unacceptable model range (Moriasi et al., 2007) where PB values ranged from 0.31 to 0.38 and 0.15 to 0.16 for calibration and validation periods, respectively, for the three watersheds. This provides evidence to support the claim that multi-objective calibration increases confidence in the model's predictive capabilities compared to using a single-objective calibration routine.

5. Conclusion

The powerful Non-Dominated Sorting Genetic Algorithm II (NSGA-II) is a popular multi-objective optimization genetic algorithm (MOGA) that has been shown to be effective for calibrating watershed models including SWAT. Because there is no known open source and freely-available software for linking NSGA-II with SWAT for model calibration, we created an open source NSGA-II/SWAT library using the Python programming language. We designed the library to be used either as a standalone tool for those experienced with Python, or as a library that can be incorporated by developers into existing third-party Graphical User Interface (GUI) software tools. In particular, a design goal was to allow for easy integration of the NSGA-II/SWAT library with the widely used SWAT-CUP program that includes many algorithms for calibrating SWAT models, but currently does not include the NSGA-II algorithm.

We demonstrated how the NSGA-II/SWAT library could be used through a test case application for calibrating a SWAT model of the Upper Neuse Watershed in North Carolina. The test case considered six objective functions: maximize Nash-Sutcliffe (E) and minimize Percent Bias (PB) as the fitness coefficients for three streamflow stations located in the watershed. Six model parameters were used in the calibration based on results obtained from using the GLUE sensitivity analysis procedure. Results from applying the NSGA-II/SWAT library to this test case showed large tradeoffs between fitness coefficients in the study watershed as illustrated in the Pareto front. In general, the Eno watershed had lower E values compare to the other two watersheds, and we suspect that this is due to urbanization within the Eno watershed that is not present in the other two watersheds.

We chose the optimal parameter set from the Pareto front when weighting all objective functions equally and used this parameter set to create the calibrated SWAT model. Results from running the calibrated SWAT model during the time period used to calibrate the model were E values ranging between 0.65 and 0.75 and PB values ranging between 0.02 and 0.08 for the three streamflow stations used for calibration. The results from running the model during an independent evaluation period not used for calibrating the model showed E values ranging between 0.59 and 0.62 and PB values ranging between -0.13 and -0.09. All results for the calibration and evaluation periods were considered to have satisfactory performance (Moriasi et al., 2007) and improved results obtained from executing the SWAT model using an optimal parameter set generated when considering only one of the six objective functions. Therefore, the model calibration resulting from using the NSGA-II/SWAT library resulted in a well-calibrated SWAT model that increases our confidence in the model's predictive capabilities compared to the more common approach of using a single objective function.

The NSGA-II/SWAT tool was written to allow for easy expansion to include other calibration algorithms and interfaces for other hydrological and environmental models that might require multi-objective calibration. By having the source code in a public repository, the code can be easily obtained and extended by others to include these enhancements. Furthermore, the software was designed in a way so that it can be easily incorporated into front-end Graphical User Interface (GUI) software tools, most notably SWAT-CUP. A proof-of-concept for incorporating the library into SWAT-CUP was shown that leverages the existing data visualization capabilities already available through SWAT-CUP and provides a new and powerful calibration routine to SWAT-CUP users. Future work could formalize the proof-of-concept by extending SWAT-CUP to accept 3rd party calibration routines as plug-ins. This would encourage adoption of new calibration algorithms more quickly and easily into the community-supported and widely used SWAT-CUP.

Acknowledgments

This work was funded in part by the US National Science Foundation under the award CBET:0846244 and by the National Oceanic and Atmospheric Administration (NOAA) Global Interoperability Program and the NOAA Environmental Software Infrastructure and Interoperability Group.

References

Abbaspour, K.C., 2013. SWAT-CUP 2012: SWAT calibration and uncertainty programs-A user manual, in: Swiss Federal Institute of Aquatic Science and Technology, Eawag.

Abbaspour, K.C., Vejdani, M., Haghighat, S., 2007. SWAT-CUP calibration and uncertainty programs for SWAT, in: MODSIM 2007 International Congress on Modelling and Simulation, Modelling and Simulation Society of Australia and New Zealand.

Andersen, J., Refsgaard, J.C., Jensen, K.H., 2001. Distributed hydrological modelling of the Senegal River Basin---model construction and validation. J. Hydrol. 247, 200–214.

Arabi, M., Govindaraju, R.S., Hantush, M.M., 2006. Cost-effective allocation of watershed management practices using a genetic algorithm. Water Resour. Res. 42.

Arnold, J.G., Srinivasan, R., Muttiah, R.S., Williams, J.R., 1998. Large area hydrologic modeling and assessment part I: Model development1. JAWRA J. Am. Water Resour. Assoc. 34, 73–89.

Arnold, J.G., Srinivasan, R., Ramanarayanan, T.S., DiLuzio, M., 1999. Water resources of the Texas gulf basin. Water Sci. Technol. 39, 121–133.

Bekele, E.G., Nicklow, J.W., 2007. Multi-objective automatic calibration of SWAT using NSGA-II. J. Hydrol. 341, 165–176.

Beven, K., Binley, A., 1992. The future of distributed models: model calibration and uncertainty prediction. Hydrol. Process. 6, 279–298.

Confesor, R.B., Whittaker, G.W., 2007. Automatic Calibration of Hydrologic Models With Multi-Objective Evolutionary Algorithm and Pareto Optimization. JAWRA J. Am. Water Resour. Assoc. 43, 981–989.

Cools, J., Broekx, S., Vandenberghe, V., Sels, H., Meynaerts, E., Vercaemst, P., Seuntjens, P., Van Hulle, S., Wustenberghs, H., Bauwens, W., others, 2011. Coupling a hydrological water quality model and an economic optimization model to set up a cost-effective emission reduction scenario for nitrogen. Environ. Model. Softw. 26, 44–51.

Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. Evol. Comput. IEEE Trans. 6, 182–197.

Ercan, M.B., Goodall, J.L., 2012. Estimating Watershed-Scale Precipitation by Combining Gauge and Radar Derived Observations. J. Hydrol. Eng. 120807052807006. doi:10.1061/(ASCE)HE.1943-5584.0000687

Gassman, P.W., Reyes, M.R., Green, C.H., Arnold, J.G., 2007. The soil and water assessment tool: Historical development, applications, and future research directions. Trans. ASABE 50, 1211–1250.

Haupt, R.L., Haupt, S.E., 2004. Practical genetic algorithms. John Wiley & Sons.

Hejazi, M., Cai, X., Borah, D., 2008. Calibrating a watershed simulation model involving human interference: an application of multi-objective genetic algorithms. J. Hydroinformatics 10, 97–111.

Kayastha, N., Shenlang, L., Betrie, G.D., Zakayo, Z., Griensven, A. van, Solomatine, D.P., 2011. Dynamic linking of the watershed model SWAT to the multi-objective optimization tool NSGAX, in: 8th IWA Symposium on Systems Analysis and Integrated Assessment, June 20-22, San Sebastian, Spain.

Khu, S.T., Madsen, H., 2005. Multiobjective calibration with Pareto preference ordering: An application to rainfall-runoff model calibration. Water Resour. Res. 41.

Kirsch, K., Kirsch, A., Arnold, J.G., 2002. Predicting sediment and phosphorus loads in the Rock River basin using SWAT. Forest 971, 10.

Liu, R., Zhang, P., Wang, X., Chen, Y., Shen, Z., 2013. Assessment of effects of best management practices on agricultural non-point source pollution in Xiangxi River watershed. Agric. Water Manag. 117, 9–18.

Lu, S., Kayastha, N., Thodsen, H., van Griensven, A., Andersen, H.E., 2014. Multiobjective calibration for comparing channel sediment routing models in the soil and water assessment tool. J. Environ. Qual. 43, 110–20. doi:10.2134/jeq2011.0364

Moriasi, D.N., Arnold, J.G., Van Liew, M.W., Bingner, R.L., Harmel, R.D., Veith, T.L., 2007. Model evaluation guidelines for systematic quantification of accuracy in watershed simulations. Trans. ASABE 50, 885–900.

Nicklow, J.W., Muleta, M.K., 2001. Watershed management technique to control sediment yield in agriculturally dominated areas. Water Int. 26, 435–443.

Niraula, R., Norman, L.M., Meixner, T., Callegary, J.B., 2012. Multi-gauge Calibration for modeling the Semi-Arid Santa Cruz Watershed in Arizona-Mexico Border Area Using SWAT. Air, Soil Water Res.

Santhi, C., Arnold, J.G., Williams, J.R., Hauck, L.M., Dugas, W.A., 2001. Application of a watershed model to evaluate management effects on point and nonpoint source pollution. Trans. ASAE 44, 1559–1570.

Shafii, M., Smedt, F. De, 2009. Multi-objective calibration of a distributed hydrological model (WetSpa) using a genetic algorithm. Hydrol. Earth Syst. Sci. 13, 2137–2149.

Srinivas, N., Deb, K., 1994. Muiltiobjective optimization using nondominated sorting in genetic algorithms. Evol. Comput. 2, 221–248.

Tang, Y., Reed, P., Wagener, T., others, 2006. How effective and efficient are multiobjective evolutionary algorithms at hydrologic model calibration? Hydrol. Earth Syst. Sci. Discuss. 10, 289–307.

Tarboton, D.G., Horsburgh, J.S., Maidment, D.R., Whiteaker, T., Zaslavsky, I., Piasecki, M., Goodall, J., Valentine, D., Whitenack, T., 2009. Development of a community hydrologic information system, in: 18th World IMACS Congress and MODSIM09 International Congress on Modelling and Simulation, Ed. RS Anderssen, RD Braddock and LTH Newham, Modelling and Simulation Society of Australia and New Zealand and International Association for Mathematics and Comput. pp. 988–994.

White, K.L., Chaubey, I., 2005. Sensitivity analysis, calibration, and validations for a multisite and multivariable swat model1. JAWRA J. Am. Water Resour. Assoc. 41, 1077–1089.

Winston, W.L., Venkataramanan, M., Goldberg, J.B., 2003. Introduction to mathematical programming. Thomson/Brooks/Cole.

Zhang, X., Beeson, P., Link, R., Manowitz, D., Izaurralde, R.C., Sadeghi, A., Thomson, A.M., Sahajpal, R., Srinivasan, R., Arnold, J.G., 2013. Efficient multi-objective calibration of a computationally intensive hydrologic model with parallel computing software in Python. Environ. Model. Softw. 46, 208–218. doi:10.1016/j.envsoft.2013.03.013

Zhang, X., Izaurralde, R.C., Zong, Z., Zhao, K., Thomson, A.M., 2012a. EVALUATING THE EFFICIENCY OF A MULTI-CORE AWARE MULTI-OBJECTIVE OPTIMIZATION TOOL FOR CALIBRATING THE SWAT MODEL. Trans. ASABE, 55(5)1723-1731.

Zhang, X., Izaurralde, R.C., Zong, Z., Zhao, K., Thomson, A.M., 2012b. EVALUATING THE EFFICIENCY OF A MULTI-CORE AWARE MULTI-OBJECTIVE OPTIMIZATION TOOL FOR CALIBRATING THE SWAT MODEL. Trans. ASABE, 55(5)1723-1731.

Zitzler, E., Deb, K., Thiele, L., 2000. Comparison of multiobjective evolutionary algorithms: Empirical results. Evol. Comput. 8, 173–195.