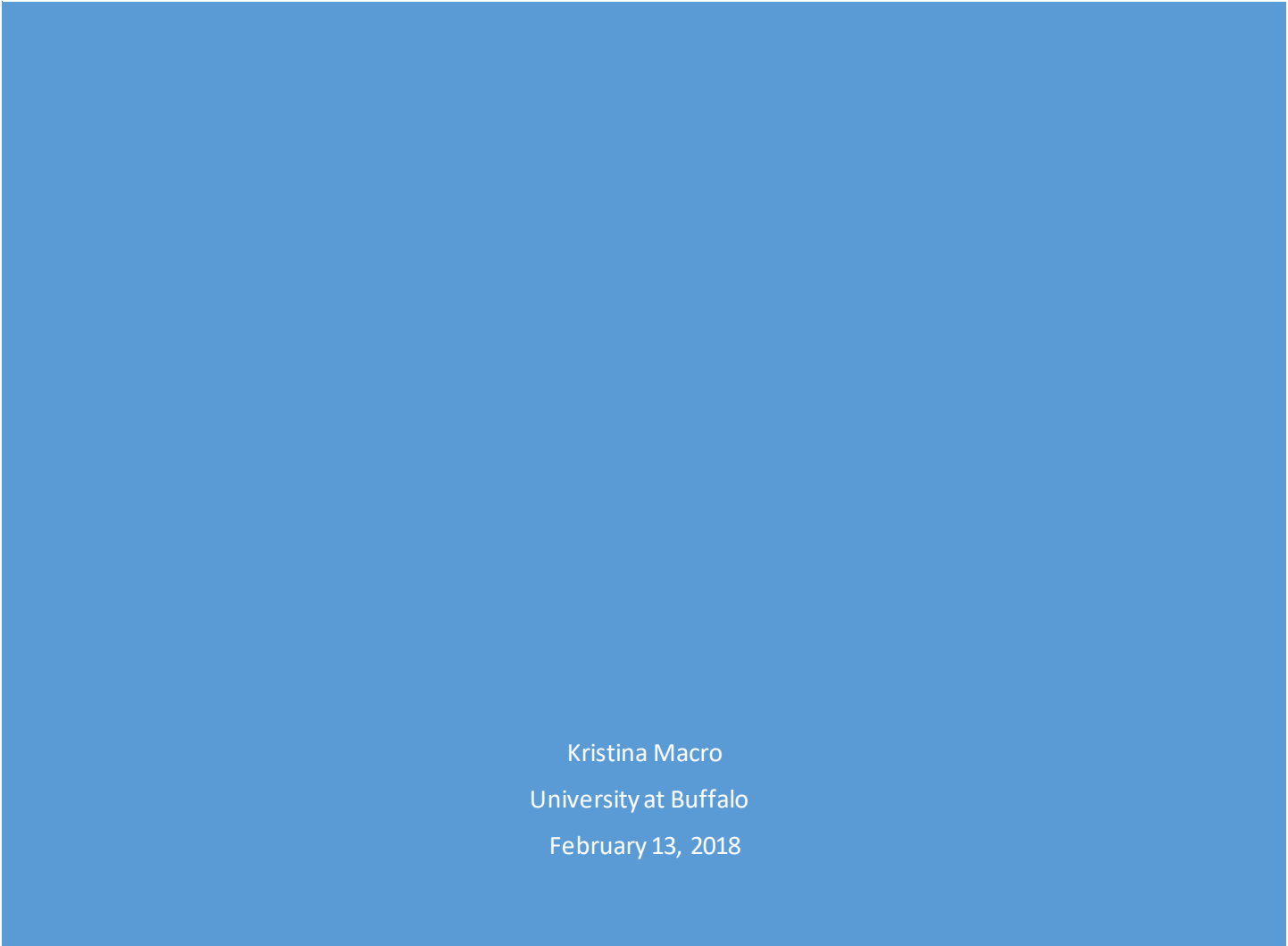




OSTRICH-SWMM

AN INTRODUCTION & TUTORIAL



Kristina Macro
University at Buffalo
February 13, 2018

Table of Contents

1. Introduction to OSTRICH-SWMM.....	1
2. Problem Introduction	1
3. OSTRICH-SWMM Inputs.....	2
3.1 Model Template.....	2
3.2 OSTRICH Input.....	3
3.3 Input Parameters	5
3.4 OSTRICH-SWMM Configuration.....	5
3.5 Input Checklist	5
4. Running OSTRICH-SWMM.....	6
4. Analyzing Output.....	6
 Figure 1 OSTRICH-SWMM Processes Flow Chart.....	1
Figure 2 Tutorial Study Area	2
Figure 3 Node J2 Total Inflow and Flooding	2
Figure 4 Node J2 Total Inflow with LIDs added by OSTRICH-SWMM.....	8
 Table 1 Response Variables	4
Table 2 OSTRICH-SWMM File Descriptions.....	6
Table 3 Optimal Solution	6
Table 4 Model_results.csv file	7

1. Introduction to OSTRICH-SWMM

OSTRICH-SWMM is a python code package that connects the operations of the Optimization Software Toolkit for Computational Heuristics (OSTRICH) with Stormwater Management Model (SWMM) inputs and outputs. This connection allows for the optimization of the size, type, and location Low Impact Development (LID) projects in a particular model. OSTRICH-SWMM is a flexible, open source tool that can utilize parallel processing and conduct either single- or multi-objective optimization using a wide variety of optimization algorithms. Figure 1 shows how OSTRICH-SWMM works. Each portion of this flow chart will be explained through a tutorial example in the following sections.

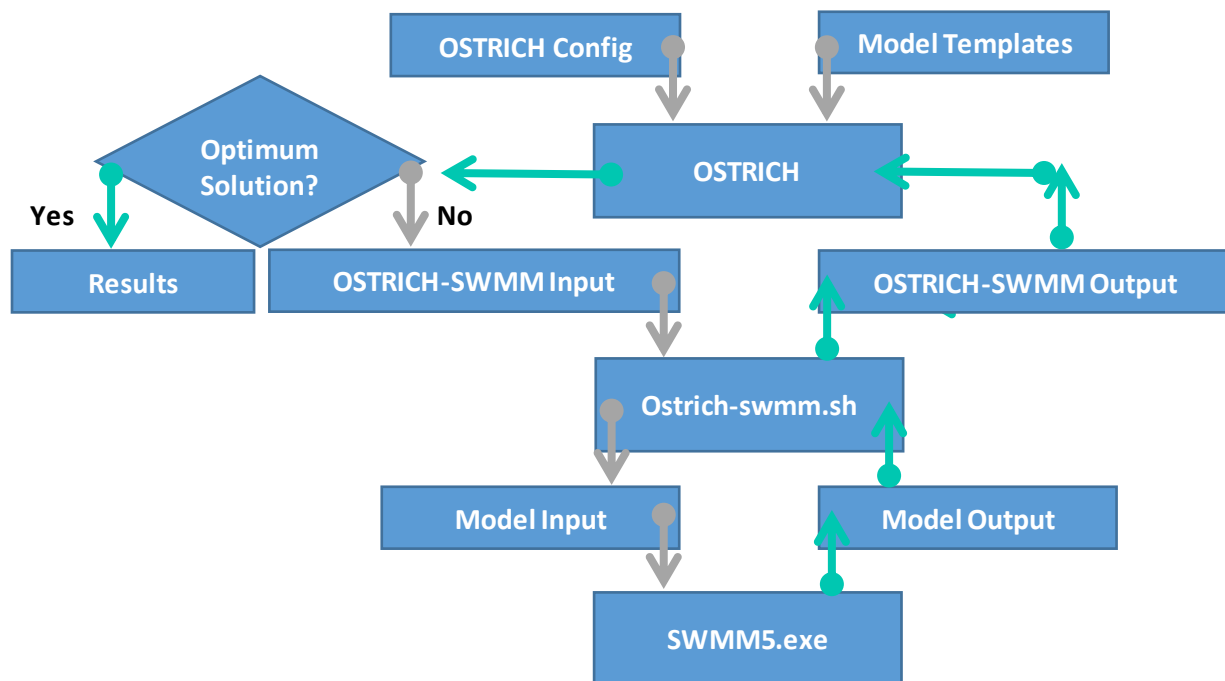


Figure 1 OSTRICH-SWMM Processes Flow Chart

2. Problem Introduction

Although OSTRICH-SWMM can be used for optimizing LID projects in complex SWMM models, a simple model from the Quick-Start Tutorial in the EPA's Storm Water Management Model User's Manual Version 5.1 will be used to demonstrate the capabilities of OSTRICH-SWMM. If you are unfamiliar with EPA SWMM or OSTRICH, it is highly recommended that you review those tutorials/manuals first before proceeding with this example. The User's Manual and other resources regarding EPA SWMM can be found at <https://www.epa.gov/water-research/storm-water-management-model-swmm>. Resources for OSTRICH can be found at <http://www.eng.buffalo.edu/~lsmatott/Ostrich/OstrichMain.html>.

The example SWMM model can be found in the file named TutorialModel.inp. The study area is shown in Figure 1. Subcatchment S3 has a lower percent impervious compared to the other subcatchments (25% compared to 50%), while subcatchment S2 is slightly larger than the other two (5 ac instead of 4 ac).

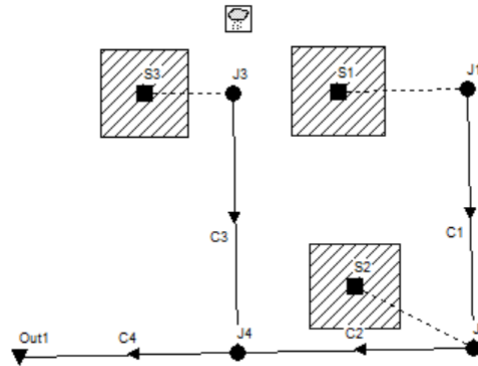


Figure 2 Tutorial Study Area

Although there is no CSO outfall in this example, flooding during a 3-inch, 6-hour design storm can be observed in node J2. This is shown in Figure 2, where it appears that flooding begins when the total inflow for node J2 goes above 2 cfs. In this example, OSTRICH-SWMM will be used to determine where and how many rain barrels need to be installed to eliminate this flooding problem.

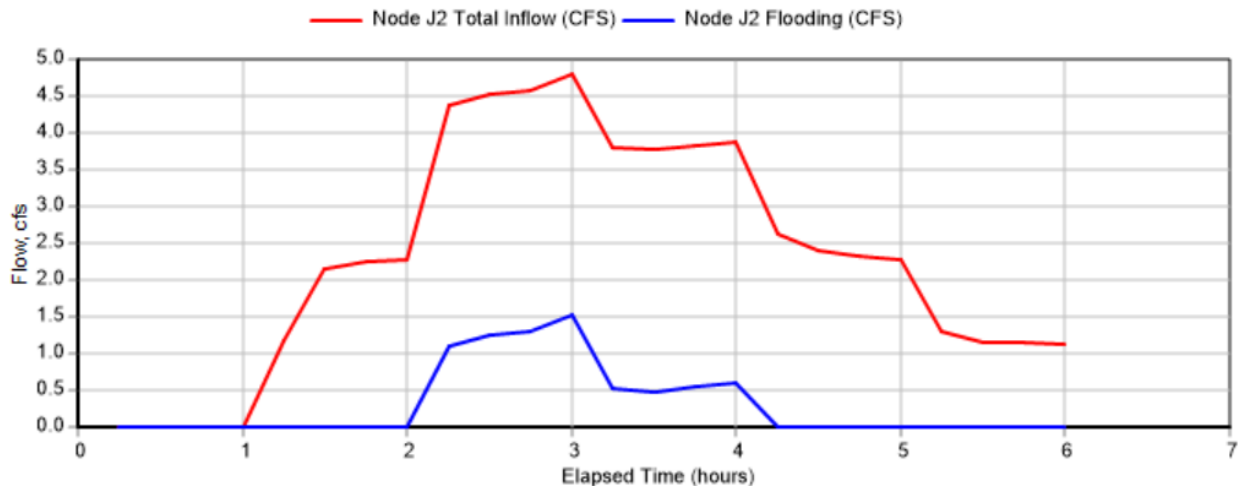


Figure 3 Node J2 Total Inflow and Flooding

3. OSTRICH-SWMM Inputs

3.1 Model Template

To begin using OSTRICH-SWMM with this example model, the model input file needs to be edited so that LID components can be added in. Save TutorialModel.inp as ModelTemplate.inp and add the following lines after the INFILTRATION section using a text editor:

[LID_CONTROLS]						
;; Type/Layer Parameters						
;;-----						
RB1	RB					
RB1	STORAGE	36	1.0	0	0	
RB1	DRAIN	58.5	0.5	0	6	
RB2	RB					
RB2	STORAGE	48	1.0	0	0	
RB2	DRAIN	26	0.5	0	6	
[LID_USAGE]						
;;Subcatchment LID Process Number Area Width InitSatur FromImprv ToPerv Report File Drain to						
;;-----						

Figure 4. LID parameter set up in SWMM template file

In this case, two sizes of rain barrels, 75 gal (RB1) and 200 gal (RB2), were added as LID controls. More information about the storage and drain parameters for different LID controls can be found in the EPA SWMM User's Manual. The LID_USAGE section is blank for now, but will be filled in by OSTRICH-SWMM as it adds LIDs to the model.

3.2 OSTRICH Input

The ostIn.txt file is the main input for OSTRICH that defines the optimization problem to be solved by OSTRICH-SWMM. The OSTRICH manual can be used to learn more about the parameters required for different types of algorithms. In this example, we will use the Dynamically Dimensioned Search (DDS) algorithm to optimize the number and location of rain barrels in the example SWMM model. To give OSTRICH the information it needs to run this algorithm, use the following steps.

General Problem Framework

- Set the Program Type to DDS
- Set the Model Executable to ostrich-swmm.sh – this is the program that OSTRICH will run to solve the optimization problem
- Set the Objective Function as the General Constrained Optimization Platform (GCOP)

File Associations

- Any files associated with the model need to be included in the ostIn file to make sure it runs properly
- List the model_input_parameters.json.tpl and model_input_parameters.json files in the File Pairs section
- Include the model-ostrich-config.json and ModelTemplate.inp files in the Extra Files section. If the rainfall gauge data is in an external .dat file, that file would need to be included as well.

Optimization Parameters

- In this example, we will optimize the number of rain barrels in subcatchments S1, S2, and S3.
- Set the initial value and upper/lower bounds for the number of rain barrels in each subcatchment in the Integer Params section

- For this example, the number of rain barrels will be bounded between 0 and 100
- To associate the rain barrels with the subcatchments in the model, the subcatchment names need to be included as Combinatorial Params

Response Variables

Based on these design variables and the desired outcome, the response variables can be defined. The response variables are extracted from the OSTRICH-SWMM output for one model run. The response variables for this example are defined in Table 1.

Table 1 Response Variables

NCSO	Number of overflow events at one node
FVOL	Total flow volume of overflow events at one node
FDUR	Duration of overflow events at one node
SUM_NRB1(RB2)	Total number of RB1 or RB2 added to model
ExcessRB1(RB2)	Total number of RB1 or RB2 added by OSTRICH that didn't fit in the model

Tied Response Variables can be calculated using these response variables. For example, the Real_Cost tied response variable can be calculated as:

$$COST = 150 \sum_{i=0}^n NRB1_i + 400 \sum_{i=0}^n NRB2_i$$

Where i represents an individual subcatchment, n equals the number of subcatchments, and the weights associated with each type of rain barrel are conservative cost estimates for purchasing and installing the rain barrels.

Optimization Framework

- Select one of the response variables as the objective for GCOP
 - In this example, FVOL will be the objective
 - OSTRICH also has multi-objective optimization algorithms that could be used for more complex problems
- Choose a Penalty Function
 - The Additive Penalty Method (APM) will be used for this example. This function calculates the penalty as the sum of the constraint violations.
- Define Constraints
 - The constraints for this example include keeping the number of excess RB equal to zero, and keeping the cost of the project under \$1,000.

$$h_1 = \sum_{i=0}^n \text{ExcessRB1}_i = 0$$

$$h_2 = \sum_{i=0}^n \text{ExcessRB2}_i = 0$$

$$g_1 = COST - 1000 \leq 0$$

Algorithm Conditions

- Parameters will change based on the type of algorithm, refer to the OSTRICH Manual for token names
- For the DDS Algorithm in this example, the perturbation value and maximum number of iterations were kept at the default values.

3.3 Input Parameters

The model_parameters_input.json.tpl file sets up the format for the LID_USAGE parameters that will be filled in by OSTRICH-SWMM. Since OSTRICH-SWMM will be changing the number and location of the LIDs, the remaining parameters can be filled in according to the guidelines in the SWMM User's Manual (Appendix D Command Line SWMM) or details specific to your project. **In this case, the rain barrels are added in as new individual subcatchments.** One of the most important parameters in this template is the "area" parameter, which is the ground surface area that will be taken up by the rain barrel. It is important to note that this file should be in a dictionary format, with commas at the end of each line except the last one under each key. The properties under the main keys (i.e. "lids") need to match with the parameters.schema.json file in the ostrich-swmm package. **The same variable names from the ostIn file should be used for the "subcatchment" and "number".** For example:

```
{ "lids": [ { "width": 0, "fromImp": 1, "location": { "subcatchment": "_SUBCAT_1_"}, "area": 3.34,
  "toPerv": 1, "type": "RB1", "number": "_NRB1_1_", "initSat": 0},
  { "width": 0, "fromImp": 1, "location": { "subcatchment": "_SUBCAT_1_"}, "area": 7.07,
  "toPerv": 1, "type": "RB2", "number": "_NRB2_1_", "initSat": 0} ] }
```

For rain barrels, properties for rooftops should also be added in as their own subcatchments. This makes the optimization results more realistic by associating each rain barrel with a roof that drains into it. Similar to the "lids" section, the "number" and "subcatchment" for the roof properties are based on the ostIn design variables. The "OutID" is either RB1 or RB2, and the "area" and "width" are based on the average roof area(ft²) that would be draining to a rain barrel in that particular subcatchment.

3.4 OSTRICH-SWMM Configuration

The model-ostrich-swmm-config.json file sets up the input and output file paths for OSTRICH-SWMM and determines which statistics will be extracted from the SWMM output. It is necessary to change the file paths to match the file names in your working directory. Since we are trying to eliminate flooding at node J2 in this example, we will set the node as J2 and the event_threshold_flow_rate as 2. Normally if we were trying to eliminate CSO events, the threshold flow rate would be set at 0 cfs.

3.5 Input Checklist

- Create SWMM input file
- Set up LID CONTROLS and LID USAGE sections in SWMM input template file
- Create ostIn.txt file
- Create model_input_parameters.json.tpl file
- Adjust model-ostrich-swmm-config.json file
- Adjust file paths for EPA SWMM and OSTRICH-SWMM in ostrich-swmm.sh file
- Adjust python package paths in ostrich-swmm\bin\ostrich-swmm

- Add files named num_lid.csv and model_results.csv in subdirectories(i.e. mod0)

4. Running OSTRICH-SWMM

To run OSTRICH-SWMM, all of the previously mentioned files need to be in the same working directory as ostrich-swmm.sh. The OSTRICH application can be run from a command line prompt with the Ostrich.exe file. More details can be found in the OSTRICH manual. The Python code package for OSTRICH-SWMM can be found at <https://github.com/ubccr/ostrich-swmm>.

Table 2 OSTRICH-SWMM File Descriptions

File or Directory	Description
ostrich_swmm/	Contains main program code
ostrich_swmm/__main__.py	Code for running the program as a command-line script
Ostrich_swmm/inject.py	Code for adding in lids (and roofs) as their own subcatchments and adjusting the original subcatchment parameters
Ostrich_swmm/extract.py	Contains statistics to be parsed from SWMM output
ostrich_swmm/data/schemas	Contains JSON Schema files that are used to validate JSON files (such as the configuration and parameter files)
ostrich_swmm/data/schemas/parameters.schema.json	Contains definitions for different options in JSON files
ostrich_swmm/swmm	Contains code related to reading and writing SWMM input files
ostrich_swmm/swmm/input.py	Contains data indices for subcatchments and lid controls
setup.py	Controls how ostrich-swmm is deployed, specifies Python packages to use with OSTRICH-SWMM
requirements.txt	A version-locked list of Python packages that setup.py pulls in
Examples, templates	Contains example input/config files

4. Analyzing Output

OSTRICH-SWMM produces many different files describing the optimal solution. The following steps present a general framework for analyzing this output and understanding the results.

1. Read **OstOutput0.txt** – this file shows the parameters and results for each trial and the optimal parameter set. For one run of this tutorial, the following optimal solution was found:

Table 3 Optimal Solution

Parameter	Value
Objective Function	1.28E+10
_NRB1_1_	0
_NRB2_1_	0
_NRB1_2_	67
_NRB2_2_	82
_NRB1_3_	43
_NRB2_3_	57
_SUBCAT_1_	S1
_SUBCAT_2_	S2

_SUBCAT_3_	S3
------------	----

2. Select the subdirectory containing the optimal parameter set, in this case mod0
3. Under this subdirectory, go through the following files:
 - a. **OstGCOPOut.txt** – contains objective function and penalty values for each trial
 - b. LIDModel.inp – this is the new SWMM input file made by OSTRICH-SWMM, which has rain barrels and roofs added in as new subcatchments. The original subcatchment parameters have also been adjusted.
 - c. LIDModel.rpt – SWMM report file, under the flooding section for this run it says that no nodes flooded during the simulation. This means that the optimal rain barrel arrangement was able to eliminate flooding at node J2.
 - d. Model_results.csv – contains the overall statistics extracted from swmm output (Table 4)

Table 4 Model_results.csv file

node_name	J2
num_flow_events	2
total_flow_volume	24697.9
total_flow_duration	5400
first_flow_start	2016-04-18T02:15:00.001000
first_flow_end	2016-04-18T03:15:00.001000
first_flow_duration	3600
first_flow_volume	7929.349
last_flow_start	2016-04-18T03:30:00.001000
last_flow_end	2016-04-18T04:00:00.001000
last_flow_duration	1800
last_flow_volume	3688.121
max_volume_flow_start	2016-04-18T02:15:00.001000
max_volume_flow_end	2016-04-18T03:15:00.001000
max_volume_flow_duration	3600
max_volume_flow_volume	7929.349
max_duration_flow_start	2016-04-18T02:15:00.001000
max_duration_flow_end	2016-04-18T03:15:00.001000
max_duration_flow_duration	3600
max_duration_flow_volume	7929.349

- e. Num_lid.csv - List of all subcatchments and associated number of each type of lid with a sum of different lid types and the number of excess LIDs that OSTRICH tried to add in at the bottom of the list
4. Analyze time series – although OSTRICH-SWMM does not produce a time series, the swmmex command can be used to produce time series for specific nodes and links
 - a. To use swmmex, first create an LIDModel.tso file (name should be same as .out file) in the mod0 directory using the following format:

```
Name | LINKID | NODEID | TYPE | TSTART | TSTOP | TSTEP
Out1 | n/a | Out1 | FLOW,DEPTH,VELOCITY | 0 | * | 1
```

```
Link1|C1 |n/a|FLOW,DEPTH,VELOCITY|0|*|1  
Node1|n/a|J1|FLOW,DEPTH,VELOCITY|0|*|1
```

- b. If you are using a Linux terminal, change the working directory to mod0 and run the following commands

```
export PATH=.../swmm/5.1.011/iface:$PATH
```

```
swmmex ./LIDModel.tso
```

where ... represents the file path for SWMM

- c. Download swmmex output for each node/link and analyze in Excel

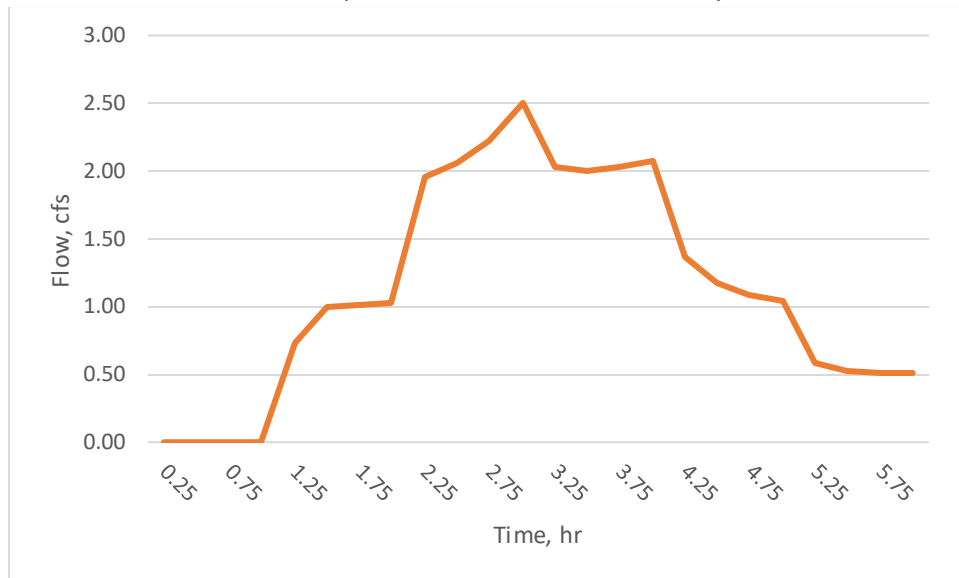


Figure 5 Node J2 Total Inflow with LIDs added by OSTRICH-SWMM

Figure 4 shows the total inflow time series for Node J2 with the optimal arrangement of rain barrels. The peak flow has been reduced from about 4.75 cfs to 2.5 cfs. This along with the model report file shows that the rain barrels made a significant impact on flooding and the overall volume of runoff entering the conduits in the example model. While this was a simple model, the number of parameters can be easily scaled up so that OSTRICH-SWMM can solve more complex optimization problems.