Sexto Congreso Nacional de Riego, Drenaje y Biosistemas
COMEII- 2021 / Hermosillo, Sonora

Artículo: COMEII-21046
Hermosillo, Son., del 9 al 11 de junio de 2021

SIMULACIÓN DE PROCESOS HIDROLÓGICOS Y ESTRUCTURAS HIDRÁULICAS CON UN ENFOQUE INTERACTIVO

SIMULATION OF HYDROLOGIC PROCESSES AND HYDRAULIC STRUCTURES USING AN INTERACTIVE APPROACH

Eduardo Jiménez Hernández[1*]

[1]Phd Student, Biosystems Engineering, University of Arizona. 1177 E. 4th Street, Shantz Building, Room 403, Tucson, AZ, USA 85721.

eduardojh@email.arizona.edu (*Corresponding author)

Abstract

In engineering is common to find computer systems for simulation of irrigation, hydraulic and hydrologic processes. This software is often offered in a specialized graphical user interface and limited by the options defined beforehand by the programmers. Programmers also determine the workflow, which typically includes data input, simulation runs, and analysis of results. However, modern interpreted programming languages such as MATLAB, R, Python or Julia, which are commonly used nowadays, allow an interactive style of simulation and analysis. This means simulations can be run step-by-step allowing changes in the simulation and model parameters, visualization and analysis of partial results, verification of intermediate calculations, and additional analysis or calculations not anticipated or provided by programmers. In this work an ongoing development of a Python 3 library is presented, which provides user friendly, flexible, and computationally efficient tools for simulation in hydrology, hydraulics, and irrigation. The library provides modules for the processing and analysis of climate data, estimation of hydrologic variables, design of hydraulic infrastructure, and more. There is no preset order in the calculations so users can define them according to their interests and needs. An advantage of this approach is the possibility of performing sensitivity analysis, uncertainty analysis, and optimization. The library is free and open-source software so it can be readily adapted to other needs and integrated to work with other Python modules or software.

Keywords: irrigation, optimization, Python, modeling

Introduction

There are software packages available for modeling and simulation of hydrologic, hydraulic, and irrigation processes, from applications developed by governments or international institutions to subscription based private applications. Popular hydraulics software includes HECRAS for river analysis and open channel flow (USACE, 2020), and EPANET a computer program that performs extended period simulation of hydraulic and water quality behavior within pressurized pipe networks (Rossman, 2000). On the other side, the Hydrologic Modeling System (HEC-HMS) which is designed to simulate the complete hydrologic processes of dendritic watershed systems (USACE, 2021), is commonly used for hydrologic modeling.

Hydraulic and hydrologic calculations are traditionally used for designing hydraulic structures or forecasting the impact of meteorological phenomena. However, sometimes the inclusion of randomness could help to increase the exploration of the solution space for these problems.

In this work a developing project for a Python 3 software library is presented, which is capable of simulate hydrologic, hydraulic, and irrigation processes. The aim of the software is to provide a flexible framework that allows further investigation of many hydraulic and hydrologic applications, with the utilization of sensitivity analysis, uncertainty analysis, and optimization.

Calculations are based on practical descriptions made in textbooks by Haan, *et al*. (1994) and Mays, (2011), which cover fundamental methods without describing the theory in detail. Due to the ambitious number of topics covered, the theoretical description of the methods utilized are beyond the scope of this work.

Materials and Methods

Structure and organization

The library is composed by several modules which include classes (object) for the processing of climate data, and calculations of hydrologic, hydraulics, irrigation, and related topics.

In Figure 1 there is a schematic representation and organization of the processes that are foundations to be covered with the software library. Hydraulic processes include the modeling and design of water flow in open channels and pressurized pipes. The most fundamental calculations of the open channel section are the geometric properties of the channel, which can be regular shapes or natural streams. These calculations can be used to design other kind of channels such as erodible, vegetated waterways, or riprap channels. Water surface profile along channels can be computed for steady flow using the standard step method, which converge to a solution using the secant method or successive substitutions. Later, these calculations can be used for hydraulic or hydrologic flow routing with Muskingum or Muskingum-Cunge methods, however these methods are not shown in the figure.

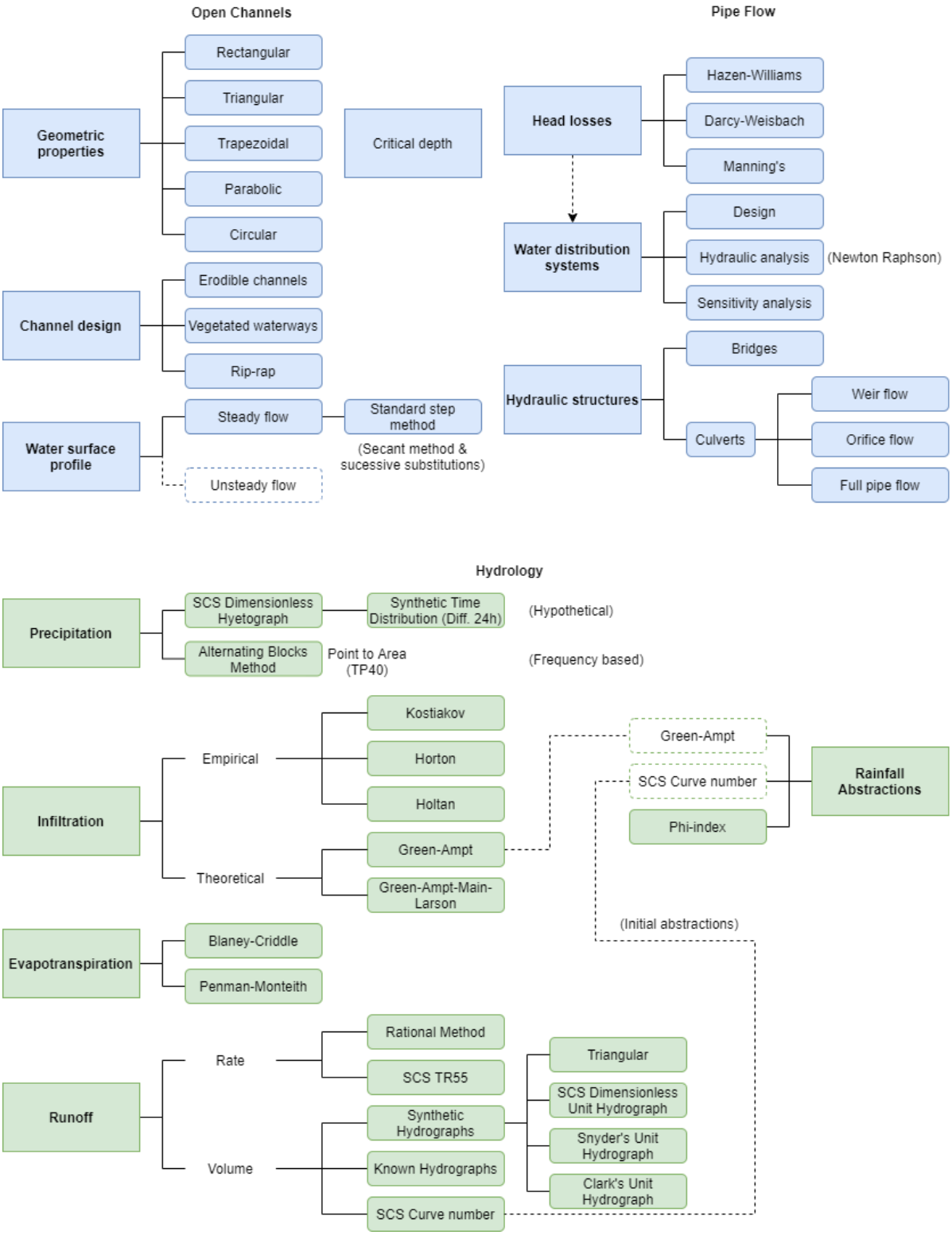

Figure 1. Schematic organization of the hydraulic and hydrologic calculations of the Python library.

The fundamental calculations for pipe flow are the head losses using Hazen-Williams, Darcy-Weisbach, and Manning's equation to design or analyze a single pipe, then it can be scaled to design or analyze a set of pipes of a larger water distribution system. Additionally, it would be interesting to analyze the change in the hydraulic parameters of the entire network due to a small change in a node or pipe, which is similar to a sensitivity analysis. The library will also contain code for modeling a wide set of hydraulic structures. The flow in pressurized pipes can be used jointly with weir and orifice flows to design or analyze structures like bridges or culverts.

Figure 1 also shows the organization of the hydrology calculations, the fundamental calculations are the precipitation analysis methods which are used to know the rainfall distribution in time, such as the alternating blocks method, or the Soil Conservation Service (SCS) dimensionless hyetograph, from which a synthetic time distribution different to a 24-hour duration can be derived. Hydrology also use losses as part of the calculations, the main losses are infiltration, evapotranspiration, and initial abstractions. All of these have different methods that can be used to estimate them. Empirical methods for estimating infiltration are Kostiakov, Horton, and Holtan equations, and theoretical equations are Green-Ampt and Green-Ampt-Main-Larson (GAML). The evapotranspiration methods are Blaney-Criddle and Penman-Monteith, both can be calculated using the common climate data obtained from the automatic weather stations. Runoff methods can be classified into rate or volumetric approaches. An example of a rate method is the rational method, examples of volume methods are synthetic and known hydrographs, or the SCS curve number. Synthetic hydrographs can be triangular, SCS dimensionless unit, Snyder's, or Clark's hydrographs. Finally, the rainfall abstractions can be estimated as infiltration methods, a fraction of the SCS curve number method, or as initial/constants losses (φ-index method). Some of these methods can be linked to carry out reservoir routing calculations which can lead to design detention ponds or other storage structures.

Results and Discussion

Preliminary results

The software library is composed of several modules which are being coded in Python 3 programming language. Python scripts are usually run from source code files directly, allowing the user to look at the parameters before running and made the necessary changes. As an interpreted language, user is free to run blocks of code without running the complete program. This allows running a program step by step and verify intermediate calculations. Having the results of a block of code, some parameter can be changed before running the next block of code. Most of the integrated development environments (IDEs) used with Python such as Spyder, Jupyter Notebook, IPython, etc., allow this interactive approach of simulation.

The project is being hosted in a GitHub repository (https://github.com/eduardo-jh/hydrx). The codename of the project is *hydrx* which is word game that suggest hydrology and hydraulics calculations are merged. The code is entirely free and open-source software, it is being released under the GNU General Public License v2.0.

The code is organized in different modules that contain classes with common methods (which are functions inside a class) to perform operations on the same data.
Among the most complete modules are the climate data processing. It is used to retrieve climate data from the Arizona Meteorological Network (AZMET), which provides records of daily or hourly data since 1987 to date for a set of automated weather stations across the state of Arizona, in the USA. The raw data is downloaded and formatted as comma separated values, then is converted into a pandas Data Frame, a convenient data set organized by columns. The user can download data for a period specified by start and end dates, or a start date and a duration in days. Figure 2 shows an example of data downloaded from the AZMET website for the period 2003-2020. An application of this feature is that users can calculate daily averages using data of two or more years.
After downloading the data, users can do interactive preprocessing, a common operation is to reduce the number of columns of the Data Set. Figure 3 (a continuation of Figure 2) shows how the user can create a list with a few column names to be selected and then apply the selection, the resulting Data Set is reduced from 28 to only 7 columns. Plots are also useful to analyze data before using it, Figure 4 shows evidence of an outlier on day 15. By reading the documentation of AZMET website it can be found that the value 999 is used as an error indicator. The user can later use proper methods to handle outliers, however, the default option is to replace them with the average of neighbor values.

```
Creating weather data for Tucson... successful!
Retrieving data for 2003 (https://cals.arizona.edu/azmet/data/0103rd.txt)... successful!
Retrieving data for 2004 (https://cals.arizona.edu/azmet/data/0104rd.txt)... successful!
Retrieving data for 2005 (https://cals.arizona.edu/azmet/data/0105rd.txt)... successful!
Retrieving data for 2006 (https://cals.arizona.edu/azmet/data/0106rd.txt)... successful!
Retrieving data for 2007 (https://cals.arizona.edu/azmet/data/0107rd.txt)... successful!
Retrieving data for 2008 (https://cals.arizona.edu/azmet/data/0108rd.txt)... successful!
Retrieving data for 2009 (https://cals.arizona.edu/azmet/data/0109rd.txt)... successful!
Retrieving data for 2010 (https://cals.arizona.edu/azmet/data/0110rd.txt)... successful!
Retrieving data for 2011 (https://cals.arizona.edu/azmet/data/0111rd.txt)... successful!
Retrieving data for 2012 (https://cals.arizona.edu/azmet/data/0112rd.txt)... successful!
Retrieving data for 2013 (https://cals.arizona.edu/azmet/data/0113rd.txt)... successful!
Retrieving data for 2014 (https://cals.arizona.edu/azmet/data/0114rd.txt)... successful!
Retrieving data for 2015 (https://cals.arizona.edu/azmet/data/0115rd.txt)... successful!
Retrieving data for 2016 (https://cals.arizona.edu/azmet/data/0116rd.txt)... successful!
Retrieving data for 2017 (https://cals.arizona.edu/azmet/data/0117rd.txt)... successful!
Retrieving data for 2018 (https://cals.arizona.edu/azmet/data/0118rd.txt)... successful!
Retrieving data for 2019 (https://cals.arizona.edu/azmet/data/0119rd.txt)... successful!
Retrieving data for 2020 (https://cals.arizona.edu/azmet/data/0120rd.txt)... successful!

In [8]: ws.data
Out[8]:
      Year  DOY  Station  TMax  ...  ET0  ET0PM  VaporPressure  DewPoint
0     2003    1        1  16.4  ...  2.1    1.8           0.51      -2.6
1     2003    2        1  20.6  ...  2.7    2.3           0.45      -4.3
2     2003    3        1  24.3  ...  2.5    2.1           0.46      -4.0
3     2003    4        1  24.3  ...  2.8    2.5           0.55      -1.5
4     2003    5        1  25.3  ...  2.5    2.2           0.59      -0.6
...    ...  ...      ...   ...  ...  ...    ...            ...       ...
6570  2020  361        1  21.6  ...  2.6    1.6           0.50      -2.7
6571  2020  362        1  24.0  ...  2.2    2.1           0.56      -1.3
6572  2020  363        1  23.4  ...  3.6    3.8           0.62       0.1
6573  2020  364        1  13.4  ...  1.8    1.6           0.70       1.7
6574  2020  365        1  15.8  ...  2.3    1.7           0.45      -4.3

[6575 rows x 28 columns]
```

Figure 2. Downloading data from the AZMET website for the period 2003-2020.

User can organize following calculations as desired, for example if the objective is to design irrigation systems, then the estimation of evapotranspiration is required (Figure 4). Although, if the user is interested in designing a hydraulic structure, then the starting point is the distribution of precipitation over time, in this case a hyetograph can be constructed using the method of the SCS Curve Type (Figure 5). Further calculations such as runoff or infiltration can be estimated if the required parameters are available or calculated by the proper functions. Once runoff is computed, for example if the structure to design is a riprap channel, there are iterative methods available to find a proper geometry to carry the design flow, and then the proper size of the riprap (Figure 6).

```
In [9]: selection =  ['Year', 'DOY', 'SR', 'TMean', 'RHMean', 'ET0', 'ET0PM']

In [10]: ws.select(selection)

In [11]: ws.data
Out[11]:
      Year  DOY     SR  TMean  RHMean  ET0  ET0PM
0     2003    1  12.35    5.8    61.2  2.1    1.8
1     2003    2  12.72    7.5    54.5  2.7    2.3
2     2003    3  12.71    9.4    48.2  2.5    2.1
3     2003    4  12.68   11.1    49.1  2.8    2.5
4     2003    5  12.57   11.4    50.5  2.5    2.2
...    ...  ...    ...    ...     ...  ...    ...
6570  2020  361  13.46    9.1    48.8  2.6    1.6
6571  2020  362  10.97   11.5    45.4  2.2    2.1
6572  2020  363  11.50   16.2    35.7  3.6    3.8
6573  2020  364   9.95   10.1    57.3  1.8    1.6
6574  2020  365  13.63    5.7    55.0  2.3    1.7

[6575 rows x 7 columns]
```

Figure 3. An example of interactive commands, the data set is reduced from 28 to only 7 columns by using a list with the column names of interest.

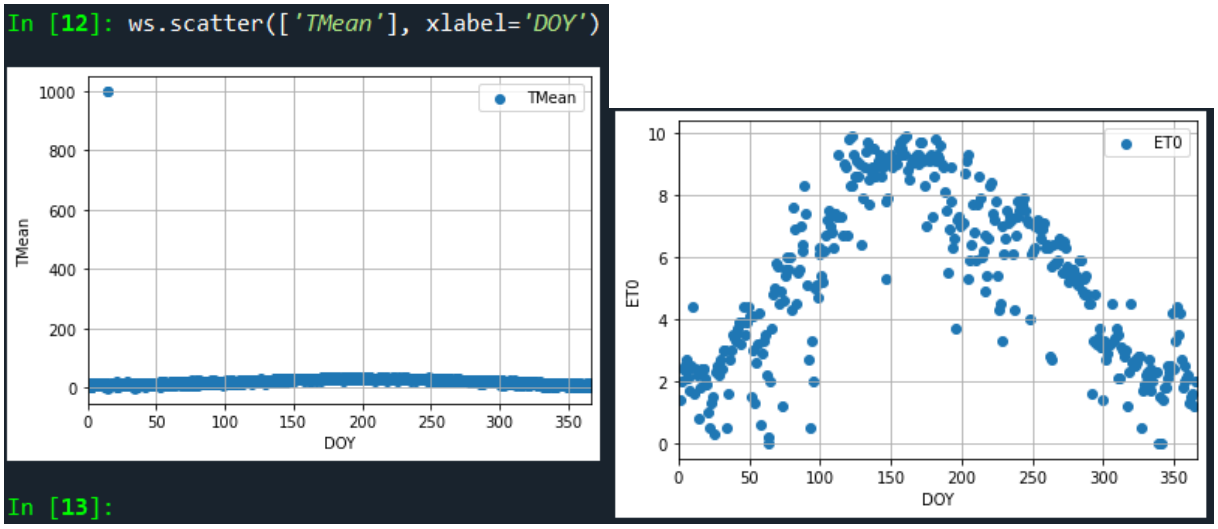

Figure 4. Plot of mean temperature (showing an outlier detected with a plot) and evapotranspiration for each day of the year.

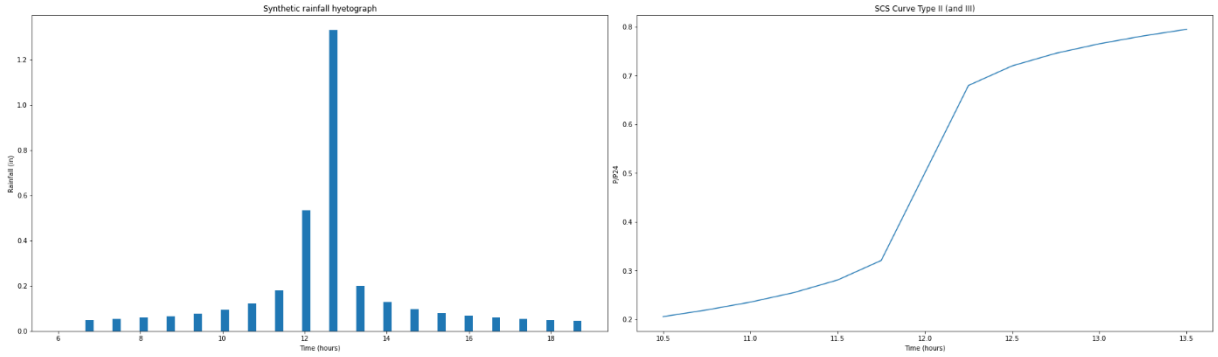

Figure 5. Basic hydrologic calculations, at the left a plot of a synthetic rainfall hyetograph for a 12-hour duration event with 40-minute timestep, and at right a plot of an approximation of SCS Curve Type II for a 3-hour duration event with 15-minute time step.

Iter	y	A	R	v	Q	Diff	Accept?
0	0.0100	0.0001	0.0035	0.1379	0.0000	30.0000	False
1	0.0200	0.0004	0.0071	0.2190	0.0001	29.9999	0
2	0.0300	0.0009	0.0106	0.2869	0.0003	29.9997	0
3	0.0400	0.0016	0.0141	0.3476	0.0006	29.9994	0
4	0.0500	0.0025	0.0177	0.4034	0.0010	29.9990	0
5	0.0600	0.0036	0.0212	0.4555	0.0016	29.9984	0
6	0.0700	0.0049	0.0247	0.5048	0.0025	29.9975	0
7	0.0800	0.0064	0.0283	0.5518	0.0035	29.9965	0
8	0.0900	0.0081	0.0318	0.5969	0.0048	29.9952	0
9	0.1000	0.0100	0.0354	0.6403	0.0064	29.9936	0
10	0.1100	0.0121	0.0389	0.6823	0.0083	29.9917	0

Iter	D	n	phi	d	tau	eta	SFb	Diff	Accept?
0	0.0100	0.0183	42.0000	0.4338	2.7070	55.2125	0.0180	1.4820	False
1	0.0200	0.0206	42.0000	0.4649	2.9013	29.5876	0.0335	1.4665	False
2	0.0300	0.0220	42.0000	0.4842	3.0213	20.5413	0.0482	1.4518	False
3	0.0400	0.0231	42.0000	0.4983	3.1095	15.8556	0.0623	1.4377	False
4	0.0500	0.0240	42.0000	0.5096	3.1797	12.9707	0.0761	1.4239	False
5	0.0600	0.0247	42.0000	0.5189	3.2382	11.0078	0.0895	1.4105	False
6	0.0700	0.0254	42.0000	0.5270	3.2885	9.5818	0.1027	1.3973	False
7	0.0800	0.0259	42.0000	0.5341	3.3327	8.4968	0.1156	1.3844	False
8	0.0900	0.0264	42.0000	0.5404	3.3722	7.6422	0.1283	1.3717	False
9	0.1000	0.0269	42.0000	0.5461	3.4079	6.9508	0.1409	1.3591	False
10	0.1100	0.0273	42.0000	0.5514	3.4405	6.3795	0.1533	1.3467	False

Figure 6. Design of a channel and riprap selection using iterative approach.

At the moment only a few modules are complete, while a large part of the code is being actively developed and hosted at the GitHub repository.

Applications in engineering

An interesting application of the Python library is the combination of different calculations that are not commonly used together. For example, analyzing a spillway requires the development of stage-discharge functions according to the inlet and outlet flow conditions. Figure 7 presents an example of a spillway analysis in three moments of the flow, the first moment at the top represents a box drop inlet that is not completely submerged by water thus working as a weir, in second figure the inlet is submerged but the pipe is still not full, in this case water behaves as orifice flow, finally at bottom figure the inlet as well as the pipe are completely full, therefore pipe flow is reached. Of course, more conditions could be present that affect the flow, such as the existence of tailwater which can be at submerged or unsubmerged conditions.

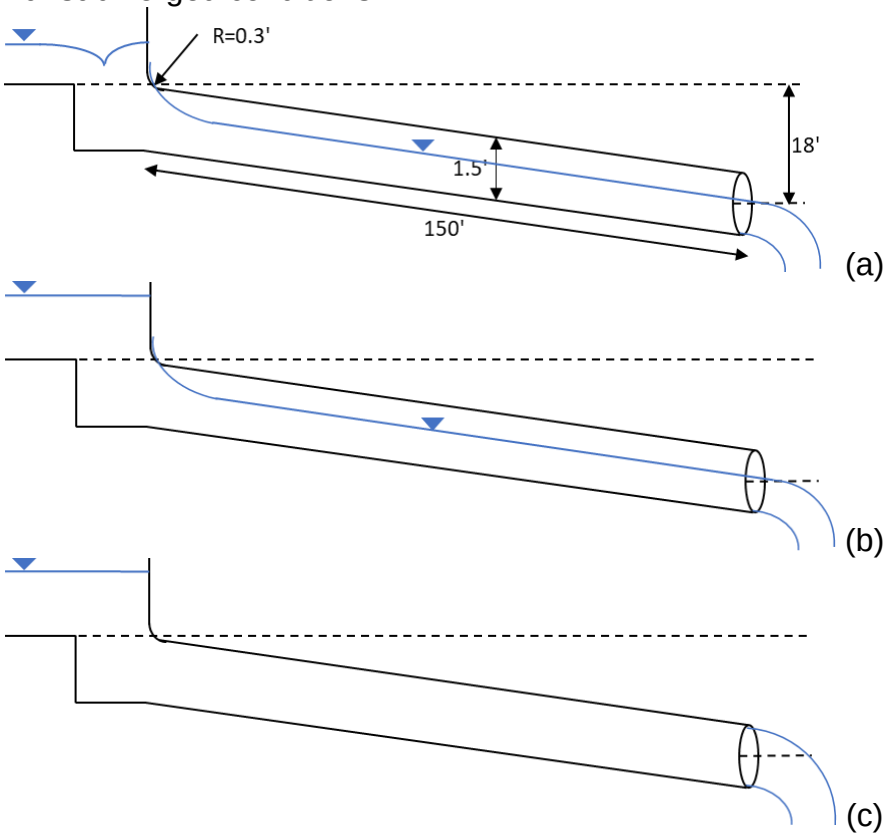

Figure 7. Schematic representation of inflow and outflow conditions for analysis in a culvert, in top figure (a) a box drop inlet works as a weir, in second figure (b) once the inlet is submerged and while the pipe is not full, it works as orifice, and finally at bottom figure (c) the pipe flow is reached.

In Figure 8 an example of a stage-discharge plot is presented. It was generated using different values of height above the crest, which correspond to the three moments described in Figure 7. It can be observed that at first flow can be modelled by a weir, however as the stage increases orifice flow is reached during a short interval, then at higher stages only pipe flow is present.

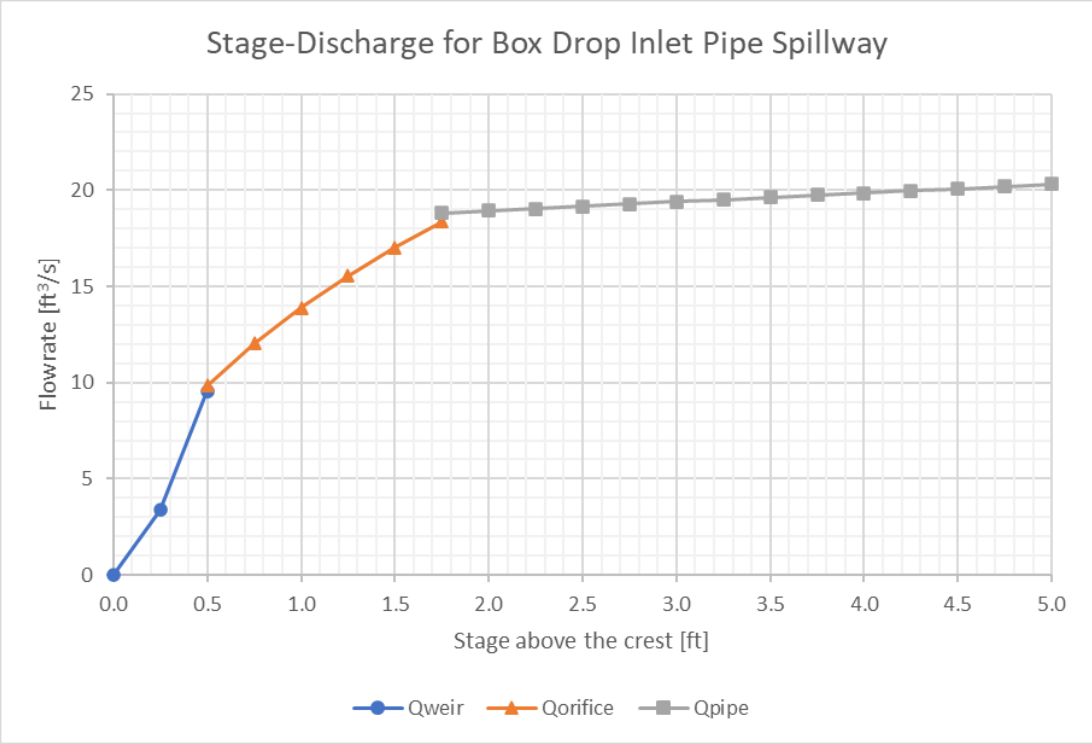

Figure 8. An example of analysis in a culvert using a stage-discharge function which combines different equations to simulate flow rate according to inflow and outflow at the three conditions presented in Figure 7.

Another example of a custom set of calculations is the analysis of detention ponds, which are a set of hydraulic structures that hold a huge amount of water in a pond for a while, and then release it slowly. This is particularly useful in urban areas, typically to avoid rapid floods from unexpected and intense rainfall. Figure 9 shows pictures of typical hydraulic structures used in a detention pond. In this case an arch concrete box shaped culvert and a weir are used to release the water. The known geometry of the structures allows us to know the flow rate at any time measuring only the height of the water.

Figure 9. Different flow conditions present in typical detention pond hydraulic structures, at the left an arch concrete box shaped culvert, at the right a detailed view of the weir above the culvert.

Additional analysis

The equations and calculations coded in the *hydrx* library are very common and there are a lot of software that can be used for design hydraulic structures and analyze hydrologic processes. However, most of these software packages are used as a tool to design or analyze a unique or specific case. The user will find obstacles if he or she wishes to perform “what-if” analysis or to include stochastic variables. Particularly if the user is dependent of packages with graphical interface, because if the programmer did not include these options, it is difficult (if not nearly impossible) to modify or adapt existing software to an application it was not designed for. To avoid the work of creating new software from scratch, this project provides tools to assemble the calculations in the order the user needs them. The drawback is that a basic Python programming knowledge is required, however it is a relatively easy programming language to learn. Most of its basic usage can be learnt in a few days or a week.

Among the additional analysis that were considered to create this project are (a) sensitivity analysis, (b) uncertainty analysis, and (c) optimization.

Sensitivity analysis is the study of how uncertainty in the output of a model (numerical or otherwise) can be apportioned to different sources of uncertainty in the model input (Saltelli et al., 2004). A related practice is ‘uncertainty analysis’, which focuses rather on quantifying uncertainty in model output. Ideally, uncertainty and sensitivity analyses should be run in tandem, with uncertainty analysis preceding in current practice (Saltelli, et al., 2008).

In a simulation problem we know the system model and some inputs, and need to compute the outputs corresponding to these inputs. In an optimization problem the model is known, together with the desired output (or a description of the desired output), and the task is to find the input(s) leading to this output (Eiben & Smith, 2015). In optimization problems the desired output is usually described as the output with best performance, which in terms of the assessment of objective variables means to find the maxima or minima. This case is illustrated with the pseudocode in Listing 1, where the objective is to find the combination of width and slope (both inputs), that results in the best performance possible (output).

Listing 1. Pseudocode for optimal design of a hydraulic structure

```
width_list = list of possible values for width
slope_list = list of possible values for slope
for each width in width_list:
    for each slope in slope_list:
        design_structure using each width and slope
        assess the performance of the designed structure
        save the performance of each structure
return best design (higher performance)
```

Conclusions

In this work an ongoing development of a software library for hydraulic and hydrologic calculations was presented. The software provides classical tools for performing calculations in a wide set of fields in irrigation, hydraulics, and hydrology. The interactive approach is a result of the combination of factors such as scripts are usually run from the source code, Python being an interpreted language, and the IDEs typically used. These factors allow the user to run code step by step, verify intermediate results, and organize the calculations as desired.

The library is flexible enough so it can be readily adapted to different methods or integrated with other Python modules to allow advanced model analysis, such as sensitivity analysis, uncertainty analysis, and optimization.

Although the current version of the software is not complete, the finished modules are fully functional and already available for downloading and testing. However, the author insists on executing intensive testing before using the software in production environments or research.

References

Eiben, A. E., & Smith, J. E. (2015). Introduction to evolutionary computing (G. Rozenberg, T. Bäck, A. E. Eiben, J. N. Kok, & H. P. Spaink (eds.); Second Edi). Springer-Verlag. https://doi.org/10.1007/978-3-662-44874-8

Haan, Barfield, Hayes, Barfield, Billy J, & Hayes, J. C. (1994). Design hydrology and sedimentology for small catchments. Academic Press.

Mays, L. W. (2011). Water resources engineering (2nd ed). John Wiley.

Rossman L. A., (2000). EPANET 2 Users Manual. United States Environmental Protection Agency. Cincinnati, OH, USA. https://nepis.epa.gov/Adobe/PDF/P1007WWU.pdf

Saltelli, A, Andres, Terry, Campolongo, Francesca, Cariboni, Jessica, Gatelli, Debora, & Ratto, Marco. (2008). Global Sensitivity Analysis (1. Aufl.). Wiley-Interscience.

US Army Corps of Engineers. 2020. HEC-RAS River Analysis System. Hydraulic Reference Manual. Version 6.0 Beta. Institute for Water Resources. Hydrologic Engineering Center. Davis CA, USA. https://www.hec.usace.army.mil/software/hec-ras/documentation/HEC-RAS_6.0_ReferenceManual.pdf

US Army Corps of Engineers. 2021. Hydrologic Modeling System (HEC-HMS) Technical Reference Manual. Hydrologic Engineering Center. Davis CA, USA. https://www.hec.usace.army.mil/confluence/hmsdocs/hmstrm