DEVELOPMENT OF A SENSITIVITY ANALYSIS APPLICATION FOR AN EVENT-BASED HEC-HMS RAINFALL-RUNOFF MODEL

Guidance document

Data4Water Summer school
Bucharest, June 20 – July 8, 2016

Andreja Jonoski, Ioana Popescu

UNESCO-IHE Institute for Water Education

Contents

1. Background to the summer school assignment

This first summer school of the Data4Water project follows the training course delivered at UPB on the topic of ‘Introduction to Hydroinformatics’. Among other topics, this training course contained a short introduction to hydrological (rainfall-runoff) modelling with the HEC-HMS modelling system developed by the United States Army Corps of Engineers (USACE).

During the exercise sessions with HEC-HMS a simple event-based rainfall-runoff model was developed consisting of only one sub-basin (the ‘Siron’ model). Next to the tools used for setting-up the model, additional support tools were briefly introduced, such as the optimization tool (for supporting model calibration) and the recently added uncertainty analysis tool for testing the influence of different parameters on the model outputs. Further to the HEC-HMS modelling system the training course also introduced the HEC-Data Storage System (HEC-DSS) of USACE, which is the database system used for most water-related USACE software packages. HEC-HMS is used in conjunction with HEC-DSS. User access to the HEC-DSS data is provided via the GUI-based application named HEC-DSSVue, which was also introduced during the training course.

The assignment for this summer school is focused on using these two software packages (HEC-HMS and HEC-DSS) for developing an additional demo application for sensitivity analysis of the outflow hydrograph to variations in the model parameters and precipitation input. The simple Siron model developed during the exercise sessions serves as basis for the summer school assignment.

Schedule

Summer school has three main components:

1. Training component: 21-23 June, 2016
2. Definition of tasks and development of work in groups: 24 June-06 July, 2016
3. Presentation of results and evaluation: 7-8 July 2016

2. Learning objectives of the summer school assignment

As most of the summer school participants are either students of computer science or have that same background, the main focus of the assignment is software development. In the assignment these participants will learn how to work with HEC-HMS and HEC-DSS, and how to extract and manipulate data coming from these packages for developing new applications. Through this process they will also become familiar with generic concepts and procedures used in hydrological rainfall-runoff modelling. Several summer school participants who have background in hydrology and other water-related topics will also benefit from the assignment because they will understand much better the inner working of HEC-HMS and HEC-DSS. Furthermore they may also develop skills for extracting and manipulating data from these systems, which can be very useful for their future hydrological modelling practice. In summary, the learning objectives of the summer school are:

1. Understanding the main concepts and procedures used in developing an event-based rainfall-runoff model with HEC-HMS.

2. Understanding the interactions between HEC-HMS and HEC-DSS in terms of data storage and exchange.
3. Development of skills for extracting data from HEC-DSS and HEC-HMS using scripting.
4. Enhancing software development skills through the design and implementation of a demo application that will use the data from HEC-HMS and HEC-DSS.

3. Introduction to the summer school assignment

The main task in the summer school assignment is to develop a demo application with a Graphical User Interface (GUI) for sensitivity analysis of the outflow hydrograph from the Siron basin to the model parameters and the precipitation inputs. The application should enable the testing of the outflow *sensitivity with respect each model parameter or input separately (one at a time).* This means that there will be no testing of combined influence (sensitivity) of two or more parameters on the outflow hydrograph. Such tasks are more complex and beyond the scope of this summer schools.

One possible GUI of the demo application is presented in Figure 1 below.

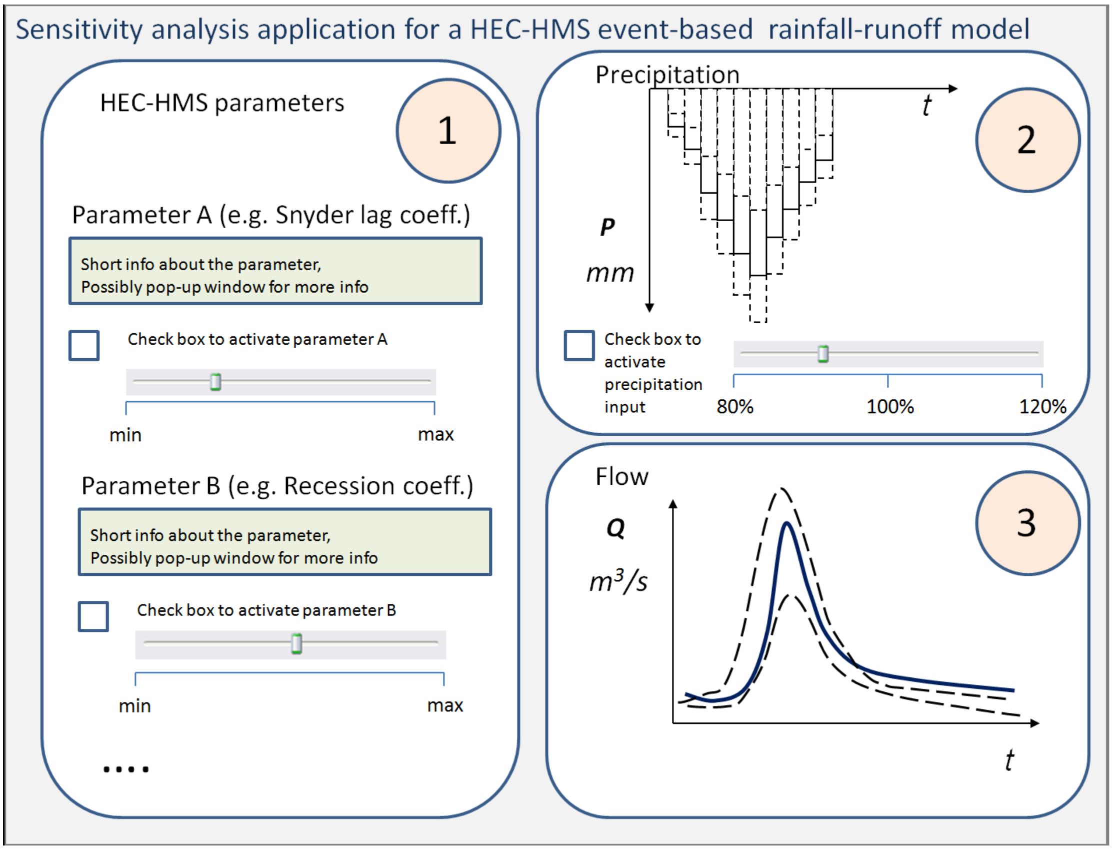

Sensitivity analysis application for a HEC-HMS event-based rainfall-runoff model
HEC-HMS parameters
1
Parameter A (e.g. Snyder lag coeff.)
Short info about the parameter,
Possibly pop-up window for more info
Check box to activate parameter A
min
max
Parameter B (e.g. Recession coeff.)
Short info about the parameter,
Possibly pop-up window for more info
Check box to activate parameter B
min
max
....
Precipitation
t
2
P
mm
Check box to activate precipitation input
80%
100%
120%
Flow
Q
m³/s
3
t

Figure 1. Possible GUI of the demo application for the summer school assignment

As can be seen in Figure 1, the demo application should consist of 3 main parts:

- Part 1 in which the users can vary the model parameters (one by one)

- Part 2 in which the users can vary the precipitation inputs (independently of any other parameters)
- Part 3 in which the influence of each model parameter or input on the outflow hydrograph can be visualised.

The basic flow hydrograph which should serve as a starting point is the one obtained from the Siron model developed during the training exercises (represented by a thick blue line in Part 3 of the GUI in the bottom right of Figure 1). This model setup is associated with certain model parameter values and certain precipitation input. The developed demo application will allow the user to vary these (in Parts 1 and 2) and see the effect on the outflow in Part 3.

Some general guidelines about the development of the three parts are provided as follows:

Part 1

As demonstrated during the training exercises, the Siron model has several parameters belonging to three different methods used for transforming the rainfall into runoff (Table 1).

Table 1 Parameters in the Siron model

Method	Parameter	Unit
Loss method	Initial Abstraction	mm
	Curve Number	-
Transform method	Snyder Lag	hr
	Snyder Cp	-
Baseflow method	Initial Baseflow	cms
	Recession Constant	n/a
	Recession Threshold	n/a

Implementing all these parameters in Part 1 of this assignment requires significant effort. Therefore it is suggested to implement three parameters in Part 1, one from each method, as in the Table 2 below:

Table 2 Parameters to be implemented in Part 1 of the summer school assignment

Method	Parameter	Unit	Range	
			Minimum	Maximum
Loss method	Curve Number	-	50	72
Transform method	Snyder Lag	hr	0.5	5
Baseflow method				
	Recession Constant	n/a	0.01	1

Any user interface elements can be implemented as long as they allow for variation of the parameters within the given range.

Part 2

The Siron model simulates the outflow hydrograph as a result of a single event (single storm). The graph representing this rainfall event is called ‘hyetograph’. The outflow hydrograph can be influenced by both the rainfall intensity (how much rainfall falls in a given period of time –for example *mm/hour*) and the shape of the hyetograph (the same rainfall amount of 100 mm in a period of 4 hours may be distributed differently: evenly each hour, most of it in the first hour, most of it in the last two ours, etc.). As indicated in Part 2 of Figure 1 (upper right), in this assignment it is suggested to vary only the rainfall intensity and not the shape of the hyetograph. This means that when the rainfall will be varied within certain range (for example ± 20%), all ordinates of the hyetograph will be varied by the same percentage. Varying the shape of the hyetograph is more complex and it is beyond the scope of this assignment.

Again, any user interface elements can be implemented as long as they allow for variation of the precipitation input within certain range.

Part 3

In this part the demo application should visualise the flow hydrograph variations. It is suggested that the basic hydrograph (from the initial model) is always shown, while another graph shows its variation in response to a parameter or precipitation input change. Thus, at least two hydrographs should be shown in the graph. Additional hydrographs can also be shown, for example corresponding to min and max values of parameters, but this is left to the choice of the participants as it may require more time and effort. Also here the choice of user interface elements is left to the participants as long as the required functionality is achieved.

Finally, the programming language and the developing environment for this assignment is up to the choice of the participants.

4. Running HEC-HMS without its GUI

HEC-HMS version 4.1 is written in Java programming language. On Windows OS it is accessible via an executable HEC-HMS.exe located in its installation folder. This executable starts the GUI of HEC-HMS through which the modelling system is normally used. Behind the GUI HEC-HMS operates with a number of input and output files, as shown in Figure 2.

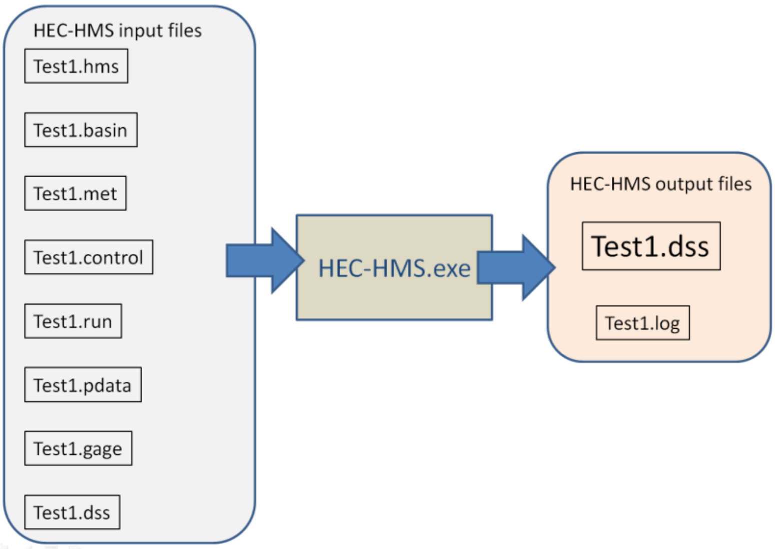
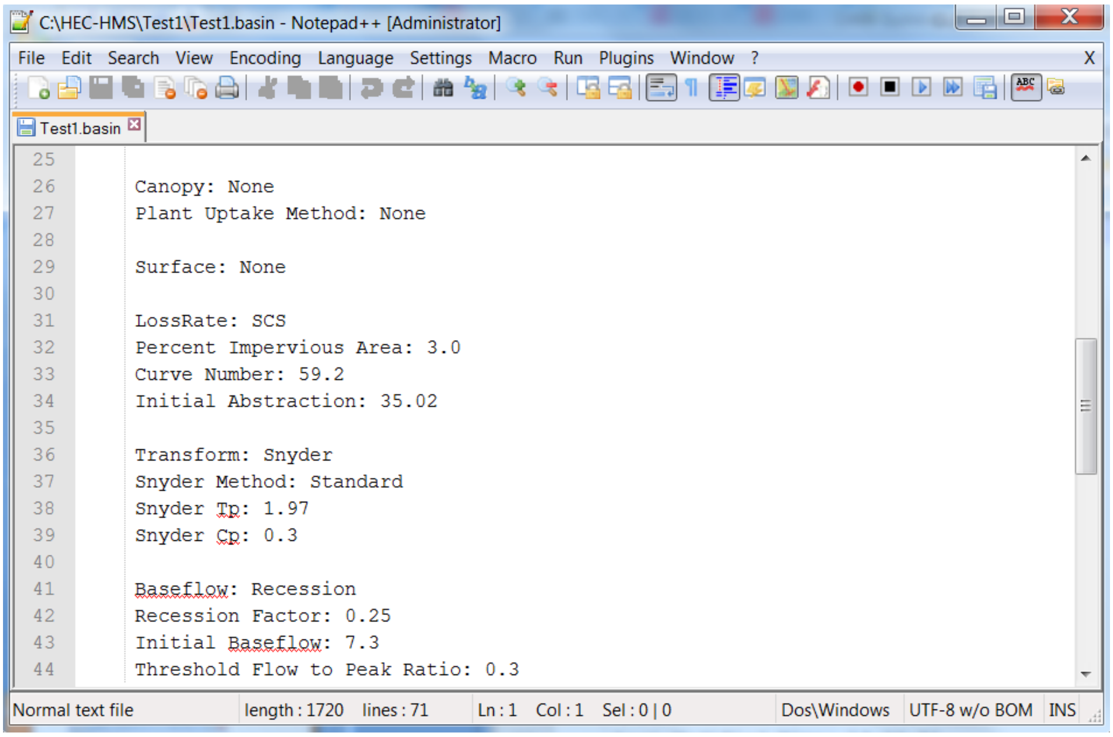


Figure 2. HEC-HMS.exe and its input and output files

The input files shown in Figure 2 are ASCII text files, which can be manipulated by the user or by an external program. For this assignment the most important input file is ‘`Test1.basin`’, which contains the data about the hydrological basin. It contains the parameter values that can be manipulated (see the screenshot in Figure 3)


C:\HEC-HMS\Test1\Test1.basin - Notepad++ [Administrator]
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
Test1.basin
Normal text file
length : 1720 lines : 71
Ln : 1 Col : 1 Sel : 0 | 0
Dos\Windows UTF-8 w/o BOM INS

```

     Canopy: None
     Plant Uptake Method: None

     Surface: None

     LossRate: SCS
     Percent Impervious Area: 3.0
     Curve Number: 59.2
     Initial Abstraction: 35.02

     Transform: Snyder
     Snyder Method: Standard
     Snyder Tp: 1.97
     Snyder Cp: 0.3

     Baseflow: Recession
     Recession Factor: 0.25
     Initial Baseflow: 7.3
     Threshold Flow to Peak Ratio: 0.3
```

Figure 3. Test1.basin input file opened in a text editor with the parameter values.

The other input files are briefly explained as follows:

- `Test1.hms` : Main configuration file of a particular HMS model. It contains pointers to other required files.
- `Test1.basin` : (see above).
- `Test1.met` : Contains data about the meteorological model used in the HMS model.
- `Test1.control` :Contains data about the control specification for a specific simulation.
- `Test1.run` : Contains data about a particular simulation run.
- `Test1.pdata` : Contains information about paired data used in the model.
- `Test1.gage` : Contains information about gage data used in the model.
- `Test1.dss` : Sometimes some inputs can be read from a HEC-DSS file (e.g. precipitation), even if the same file is used for storing outputs of the simulation.

All outputs of a HEC-HMS model are stored in a corresponding HEC-DSS file (`Test1.dss` in this example). A limited number of model outputs (summary tables, graphs) can be accessed and viewed by the user directly from the HEC-HMS interface. Access to all output data is however possible only by accessing the `Test1.dss`. These files are accessed by the HEC-DSSVue application, which enables viewing, modifying and exporting HEC-DSS files in different formats.

Note that ‘Test1’ is just a generic name used in this example. Individual controls, model runs, meteorological models, etc. can also have different names set up through the HEC-HMS GUI, so the corresponding input text files may also have different names.

If an external program needs to be developed to run HEC-HMS without the GUI it needs to basically perform the three tasks depicted in Figure 4:

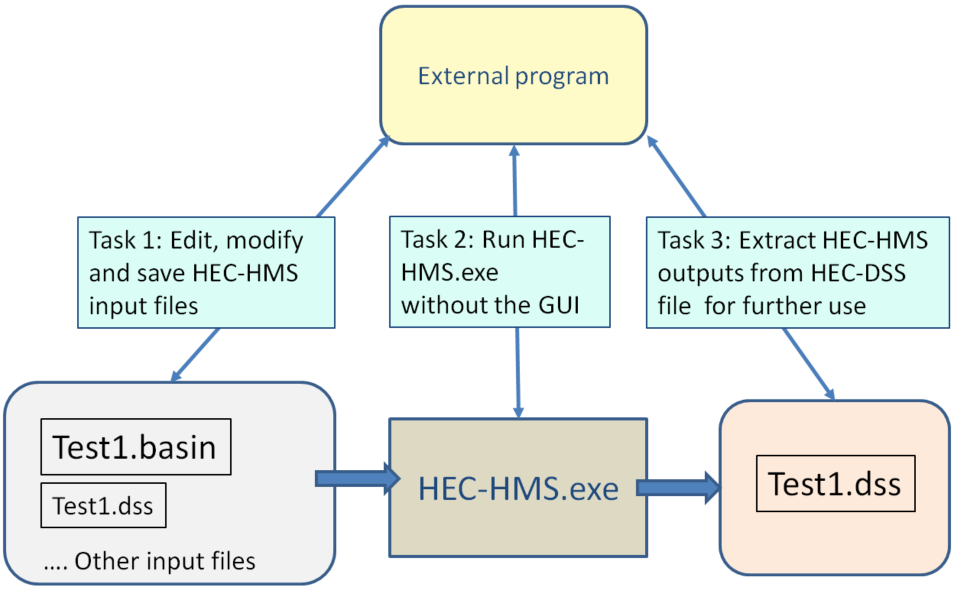

External program
Task 1: Edit, modify and save HEC-HMS input files
Task 2: Run HEC-HMS.exe without the GUI
Task 3: Extract HEC-HMS outputs from HEC-DSS file for further use
Test1.basin
Test1.dss
.... Other input files
HEC-HMS.exe
Test1.dss

Figure 4. Required tasks for running HEC-HMS without its GUI

1. The first task of editing, modifying and saving the input files can be achieved by normal file manipulation available in any programming language. Sometimes some inputs of the model need to be read from a HEC-DSS file. In that case similar operations like in task 3 below are needed.
2. Task 2 is for actual execution of a HEC-HMS model run without the GUI. There is a limited support for this, as explained in ‘Chapter 1: Introduction’ of the HEC-HMS User Manual (page 17). Basically a script (e.g. named `compute.script)` needs to be prepared which may look like the following:

   ```
   from hms.model.JythonHms import *
   OpenProject ("Test1", "C:\HEC-HMS\Test1")
   Compute ("Run1")
   Exit(1)
   ```

   Here the first line sets the scripting environment and the following lines open a particular Project, execute a particular Run and exit the programme. This script can be placed in the folder of the actual model and called by a pre-prepared batch command available in the HEC-HMS installation folder:

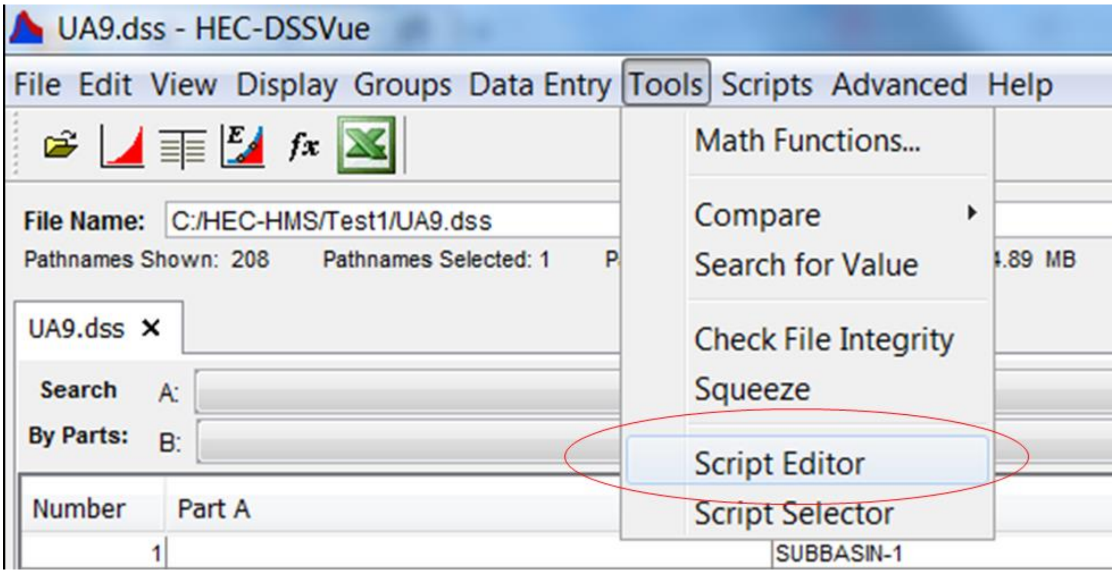
   ```
   hec-hms.cmd -s c:\hec-hms\Test1\compute.script
   ```

   This command can be put in a batch file that can be called by an external program.

3. Task 3 is about accessing the outputs of a HEC-HMS model, without the GUI of HEC-HMS or HEC-DSS. This requires that the HEC-DSS file (`Test1.dss`) is accessed programmatically. For this purpose there is Jython scripting support available in HEC-DSSVue. Users can write scripts for retrieving data stored in a HEC-DSS file, manipulating them and storing them back in the same or another HEC-DSS file, or in another output format. The next section introduces briefly the Jython scripting support available in HEC-DSSVue.

5. ACCESSING AND MANIPULATING HEC-DSS FILES

Jython is a scripting language, which is basically Python written in Java. It is designed for easy integration with Java applications. HEC-DSSVue is also written in Java. On Windows OS it is accessible via an executable HEC-DSSVue.exe located in its installation folder. This starts the GUI of the application through which the user can retrieve, manipulate, view and store HEC-DSS files. The Jython scripting support is available from within the GUI. The most important component is the 'Script editor' located under the 'Tools' menu (see Figure 5 below).



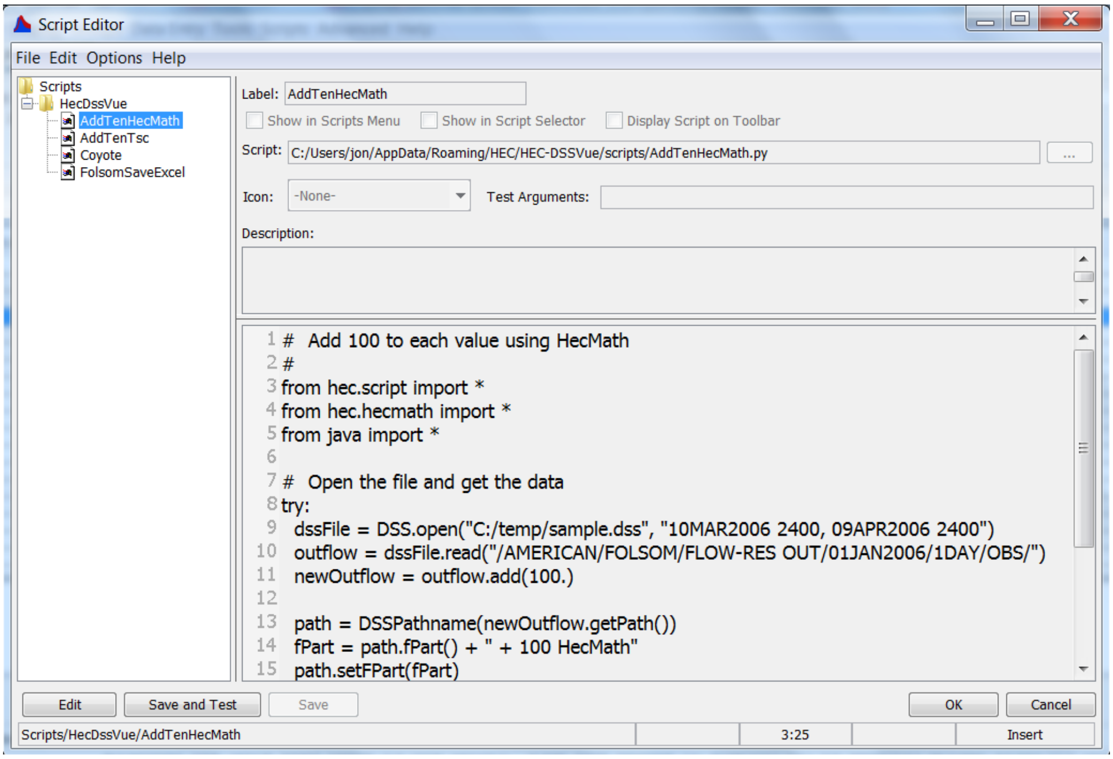
File Name: C:/HEC-HMS/Test1/UA9.dss

Pathnames Shown: 208 Pathnames Selected: 1

Figure 5. Access to the Script editor in HEC-DSSVue

The Script editor is used for importing existing scripts, creating new scripts and for executing them. For each script in the Script editor the user can select whether it should be visible in the HEC-DSSVue toolbar, in the 'Scripts' menu or in the separate window called 'Script selector', available under 'Tools' (see the upper part of Figure 6 below). All these locations just provide different ways of accessing and running the available scripts, but main script development takes place via the Script editor.

All scripts that are used are stored in a default directory. A selected script can be edited, saved and executed using the buttons located in the lower part of the Script editor (see Figure 6). While executing the scripts the user can also open a console window, to see possible console outputs generated by scripts as well as messages from the script interpreter The console window can be opened via the main menu 'Advanced'->'Console output' (see Figure 7).



Label: AddTenHecMath

□ Show in Scripts Menu □ Show in Script Selector □ Display Script on Toolbar

Script: C:/Users/jon/AppData/Roaming/HEC/HEC-DSSVue/scripts/AddTenHecMath.py

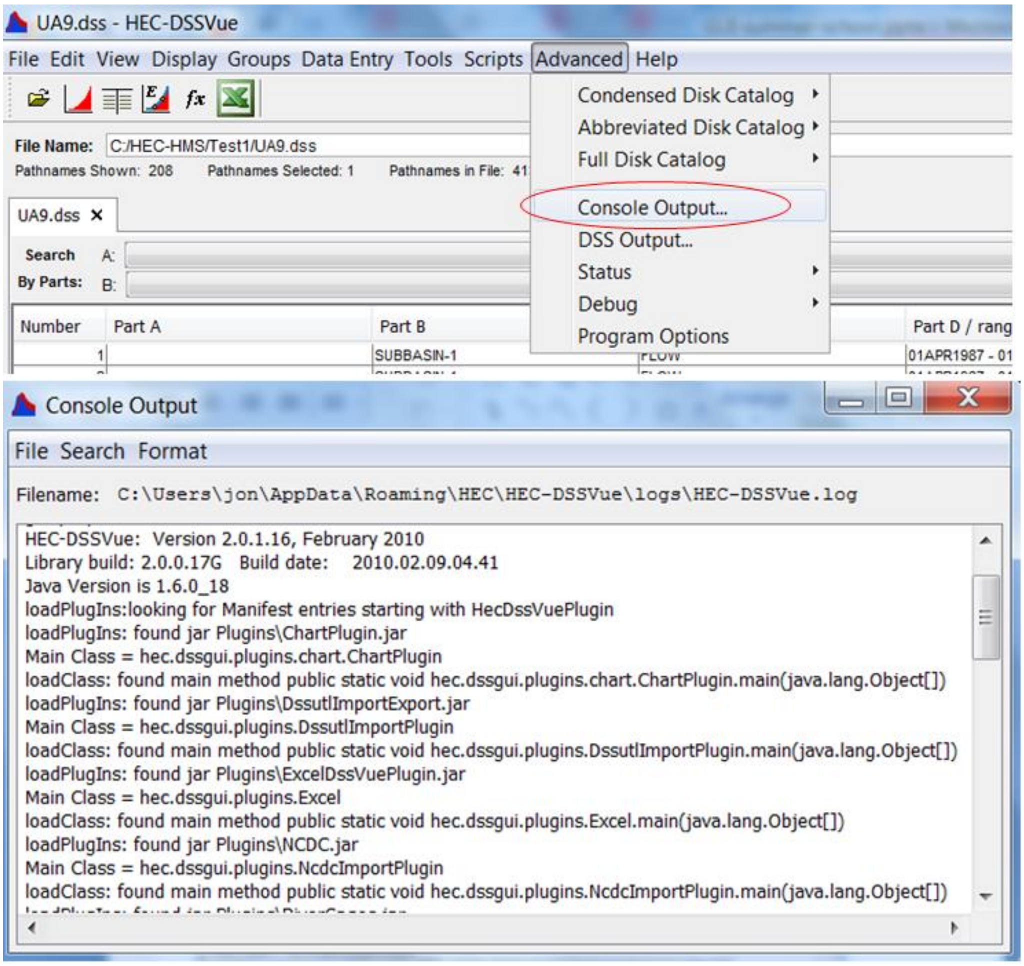
Icon: -None- Test Arguments:

Description:

```
#  Add 100 to each value using HecMath
#
from hec.script import *
from hec.hecmath import *
from java import *

#  Open the file and get the data
try:
  dssFile = DSS.open("C:/temp/sample.dss", "10MAR2006 2400, 09APR2006 2400")
  outflow = dssFile.read("/AMERICAN/FOLSOM/FLOW-RES OUT/01JAN2006/1DAY/OBS/")
  newOutflow = outflow.add(100.)

  path = DSSPathname(newOutflow.getPath())
  fPart = path.fPart() + " + 100 HecMath"
  path.setFPart(fPart)
```

Scripts/HecDssVue/AddTenHecMath 3:25 Insert

Figure 6. Script editor window of HEC-DSSVue



File Name: C:/HEC-HMS/Test1/UA9.dss

Pathnames Shown: 208 Pathnames Selected: 1 Pathnames in File: 41

Number	Part A	Part B	Part D / rang
1		SUBBASIN-1	01APR1987 - 01

Filename: `C:\Users\jon\AppData\Roaming\HEC\HEC-DSSVue\logs\HEC-DSSVue.log`

```
HEC-DSSVue:  Version 2.0.1.16, February 2010
Library build: 2.0.0.17G   Build date:    2010.02.09.04.41
Java Version is 1.6.0_18
loadPlugIns:looking for Manifest entries starting with HecDssVuePlugin
loadPlugIns: found jar Plugins\ChartPlugin.jar
Main Class = hec.dssgui.plugins.chart.ChartPlugin
loadClass: found main method public static void hec.dssgui.plugins.chart.ChartPlugin.main(java.lang.Object[])
loadPlugIns: found jar Plugins\DssutlImportExport.jar
Main Class = hec.dssgui.plugins.DssutlImportPlugin
loadClass: found main method public static void hec.dssgui.plugins.DssutlImportPlugin.main(java.lang.Object[])
loadPlugIns: found jar Plugins\ExcelDssVuePlugin.jar
Main Class = hec.dssgui.plugins.Excel
loadClass: found main method public static void hec.dssgui.plugins.Excel.main(java.lang.Object[])
loadPlugIns: found jar Plugins\NCDC.jar
Main Class = hec.dssgui.plugins.NcdcImportPlugin
loadClass: found main method public static void hec.dssgui.plugins.NcdcImportPlugin.main(java.lang.Object[])
```

Figure 7. Accessing the console output window in HEC-DSSVue

Several libraries (modules) are available for manipulating HEC-DSS files with Jython scripts. The main module is `hec.script`, and additional modules provide relevant classes for working with HEC-DSS files. Note that in Python a modulle *does not* have to be imported if the newly needed object is obtained by calling a method of an object that is already available.

The most important classes for this assignment are summarized in Table 3 below.

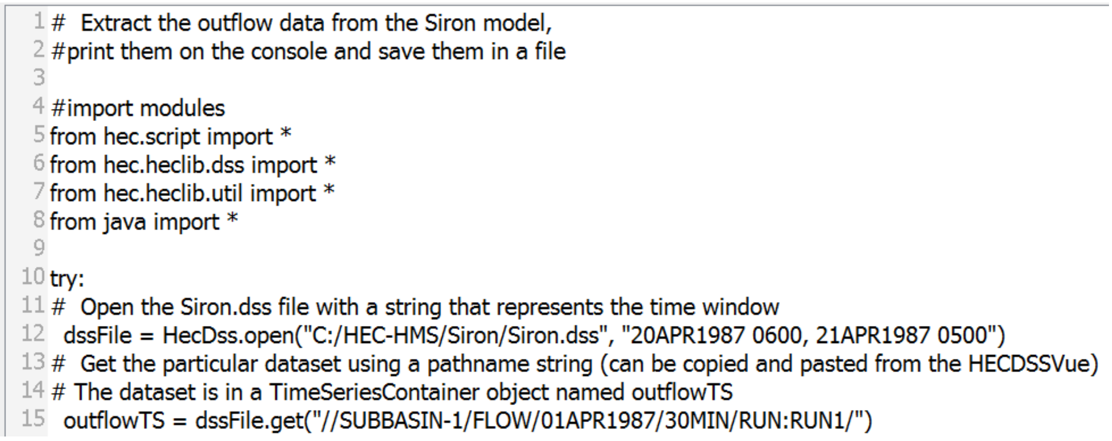
Table 3. Several main classes available in the scripting environment of HEC-DSS

Class name	Module	Purpose
`HecDss`	`hec.heclib.dss`	Main class for getting access to HEC-DSS files, with methods for opening and releasing them

`DataContainer`
`TimeSeriesContainer`
`PairedDataContainer`
	`hec.io`	Main class for getting data from a HEC-DSS file. Returned when `get()` methods are called on HecDSS objects. Depending on the type of data one of the two types of data containers below may be returned:
		`TimeSeriesContainer` for Time Series data
		`PairedDataContaine` for Paired Data
		These contain methods for actual data retrieval and manipulation.

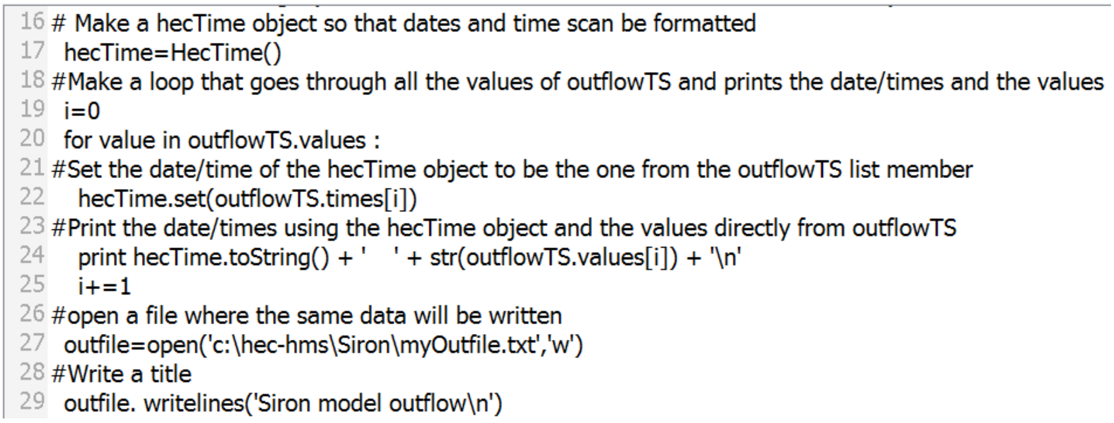
`HecTime`	`hec.heclib.util`	Utility class for creating objects that manage the date and time information
`HecMath`	`hec.hecliTb.dss`	Class that can be used as if it is a DataContainer, but has many additional methods for mathematical operations on the data. Can be created by calling methods `read()` on `HecDss` objects or from data containers, by using the method `createInstance()`, in which the DataContainer is passed as a parameter. For example: `HecMath.createInstance(myDataContainer)`
Classes for creating plots and tables	`hec.script`	There are a number of such classes for creating plots and graphs *within HEC-DSSVue*, as these are important tasks for which scripts are developed.

An example script (with detailed comments) for reading the outflow data from the Siron model developed during the training exercises is shown in Figures 8-10 below.



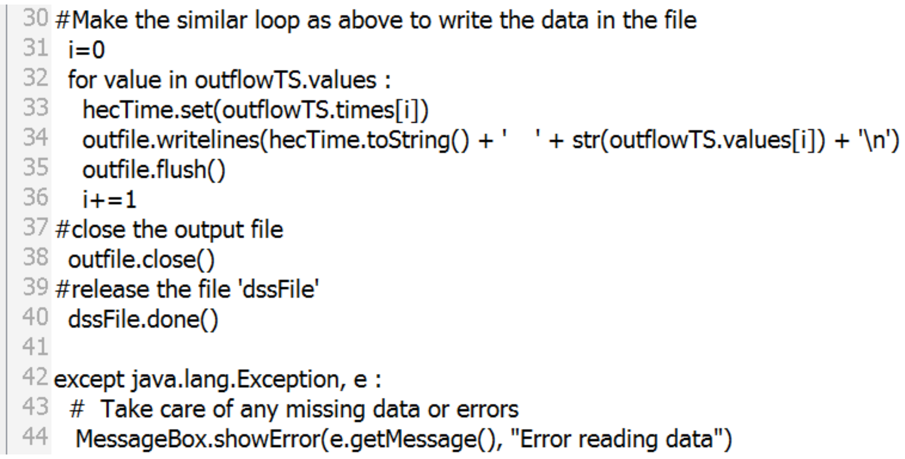
```
#  Extract the outflow data from the Siron model,
#print them on the console and save them in a file

#import modules
from hec.script import *
from hec.heclib.dss import *
from hec.heclib.util import *
from java import *

try:
#  Open the Siron.dss file with a string that represents the time window
  dssFile = HecDss.open("C:/HEC-HMS/Siron/Siron.dss", "20APR1987 0600, 21APR1987 0500")
#  Get the particular dataset using a pathname string (can be copied and pasted from the HECDSSVue)
# The dataset is in a TimeSeriesContainer object named outflowTS
  outflowTS = dssFile.get("//SUBBASIN-1/FLOW/01APR1987/30MIN/RUN:RUN1/")
```

Figure 8. Example Script for reading outflow data from the Siron HEC-DSS file (1)



```
# Make a hecTime object so that dates and time scan be formatted
 hecTime=HecTime()
#Make a loop that goes through all the values of outflowTS and prints the date/times and the values
 i=0
 for value in outflowTS.values :
#Set the date/time of the hecTime object to be the one from the outflowTS list member
   hecTime.set(outflowTS.times[i])
#Print the date/times using the hecTime object and the values directly from outflowTS
   print hecTime.toString() + '    ' + str(outflowTS.values[i]) + '\n'
   i+=1
#open a file where the same data will be written
 outfile=open('c:\hec-hms\Siron\myOutfile.txt','w')
#Write a title
 outfile. writelines('Siron model outflow\n')
```

Figure 9. Example Script for reading outflow data from the Siron HEC-DSS file (2)



```
#Make the similar loop as above to write the data in the file
 i=0
 for value in outflowTS.values :
   hecTime.set(outflowTS.times[i])
   outfile.writelines(hecTime.toString() + '    ' + str(outflowTS.values[i]) + '\n')
   outfile.flush()
   i+=1
#close the output file
 outfile.close()
#release the file 'dssFile'
 dssFile.done()

except java.lang.Exception, e :
  #  Take care of any missing data or errors
  MessageBox.showError(e.getMessage(), "Error reading data")
```

Figure 10. Example Script for reading outflow data from the Siron HEC-DSS file (3)

Detailed explanations and overview of all available classes and methods with small examples are available in Chapter 8 of the HEC-DSSVue User Manual entitled ‘Scripting’. This is the most important reference document for the assignment. Furthermore Appendix C of the same HEC-DSSVue User manual entitled ‘Example Scripts’ contains several pre-prepared scripts which can be tested. The example scripts explained in this appendix are located in the ‘samples’ folder of the HEC-DSSVue installation folder. They can be imported in the Script Editor and tested. Developing new scripts is probably easiest by starting from one of these example scripts and modifying them.

Finally, scripts can be run in non-interactive mode by calling `HEC-DSSVue.exe` and passing the script file as a parameter. For example, the above presented script, entitled ‘`ReadSironFlow.py`’ is placed in the HEC-DSS installation folder and called by:

```
HEC-DSSvue.exe C:\HEC-DSS\ReadSironFlow.py
```

This can be useful for preparing the codes needed in this assignment.

6. Guidelines for developing the demo application (sensitivity analysis of model parameters)

This section provides some guidelines for developing the demo application, mainly focusing on working with HEC-HMS and HEC-DSSVue for preparation of the necessary data needed for the application. There are no guidelines for the actual user interface development and this part is left to the participants.

One can imagine that the demo application in fact consists of two different kinds of interactions: 1) For sensitivity analysis of model parameters the interaction needs to be between Part1 and Part 3 of the demo application (see again Figure 1); 2) For sensitivity analysis of precipitation inputs the interaction needs to be between Part 2 and Part 3 of the application. The following two sub-sections will provide guidelines for generating data for these two interactions.

6.1. Guidelines for part 1 – part 3 of the demo application: sensitivity analysis of model parameters

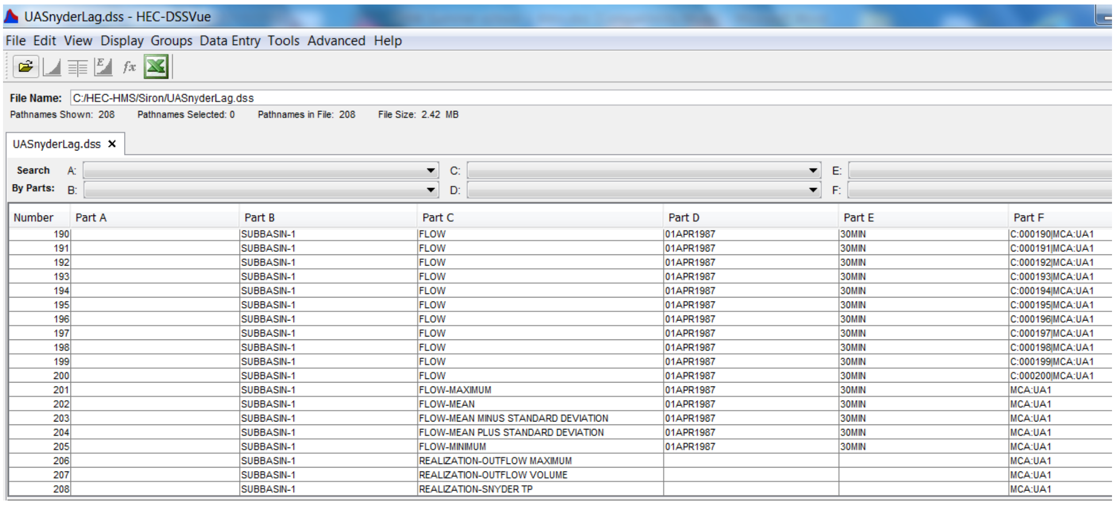
Two possible approaches are suggested for this interaction:

6.1.1. Approach 1: Using the Uncertainty Analysis Tool of HEC-HMS

Part 1 – Part 3 of the demo application is about enabling varying of individual parameter values and obtaining results in terms of changes in the outflow hydrograph. This in fact is very similar to the functionalities available in the Uncertainty Analysis tool of HEC-HMS, already demonstrated during the training exercise. In this analysis one or more parameters are sampled (many times) from a particular probability distribution and the HEC-HMS model is run for each parameter value realisation. HEC-HMS presents as results few characteristic outputs (mean, mean ± standard deviation, min, max etc). However, all results from a particular uncertainty analysis are in fact stored in a HEC-DSS file associated with it. This file can simply be used for extracting data for this application. It contains the data for all parameter realisations and the outputs (e.g. flow) for each of the parameter realisation. This means that all data can be prepared by simply running the required uncertainty analyses and then extracting the data from the corresponding HEC-DSS file.

For example, one uncertainty analysis has been run for the Siron model outflow regarding the 'Snyder lag' model parameter. The sampling was with 200 realisations from a uniform probability distribution. A HEC-DSS file named '`UACnyderLag.dss`' has been generated, which contains data for all parameter realisations and the outflow data for each realisation.

One view of this file is shown in Figure 11 below. The view shows the last part of the dataset listing, where the last few datasets (out of the 200) of the outflow results are shown, followed by few characteristic outflow results and the last dataset (number 208) contains the data for the 200 parameter realisations.



UASnyderLag.dss - HEC-DSSVue

File Edit View Display Groups Data Entry Tools Advanced Help

File Name: C:/HEC-HMS/Siron/UASnyderLag.dss

Pathnames Shown: 208 Pathnames Selected: 0 Pathnames in File: 208 File Size: 2.42 MB

UASnyderLag.dss

Search
By Parts:
A:
B:
C:
D:
E:
F:

Number	Part A	Part B	Part C	Part D	Part E	Part F
190		SUBBASIN-1	FLOW	01APR1987	30MIN	C:000190|MCA:UA1
191		SUBBASIN-1	FLOW	01APR1987	30MIN	C:000191|MCA:UA1
192		SUBBASIN-1	FLOW	01APR1987	30MIN	C:000192|MCA:UA1
193		SUBBASIN-1	FLOW	01APR1987	30MIN	C:000193|MCA:UA1
194		SUBBASIN-1	FLOW	01APR1987	30MIN	C:000194|MCA:UA1
195		SUBBASIN-1	FLOW	01APR1987	30MIN	C:000195|MCA:UA1
196		SUBBASIN-1	FLOW	01APR1987	30MIN	C:000196|MCA:UA1
197		SUBBASIN-1	FLOW	01APR1987	30MIN	C:000197|MCA:UA1
198		SUBBASIN-1	FLOW	01APR1987	30MIN	C:000198|MCA:UA1
199		SUBBASIN-1	FLOW	01APR1987	30MIN	C:000199|MCA:UA1
200		SUBBASIN-1	FLOW	01APR1987	30MIN	C:000200|MCA:UA1
201		SUBBASIN-1	FLOW-MAXIMUM	01APR1987	30MIN	MCA:UA1
202		SUBBASIN-1	FLOW-MEAN	01APR1987	30MIN	MCA:UA1
203		SUBBASIN-1	FLOW-MEAN MINUS STANDARD DEVIATION	01APR1987	30MIN	MCA:UA1
204		SUBBASIN-1	FLOW-MEAN PLUS STANDARD DEVIATION	01APR1987	30MIN	MCA:UA1
205		SUBBASIN-1	FLOW-MINIMUM	01APR1987	30MIN	MCA:UA1
206		SUBBASIN-1	REALIZATION-OUTFLOW MAXIMUM			MCA:UA1
207		SUBBASIN-1	REALIZATION-OUTFLOW VOLUME			MCA:UA1
208		SUBBASIN-1	REALIZATION-SNYDER TP			MCA:UA1

Figure 11. Uncertainty analysis result file in HEC-DSSVue.

To obtain the data needed for the Part 1 –Part 3 interaction each parameter realisation needs to be extracted from the last dataset and associated with the corresponding outflow result dataset. This can be achieved using the scripting support introduced in the previous section.

In summary the steps that may be followed in this approach are as follows:

1. For each model parameter prepare one Uncertainty analysis in HEC-HMS with sufficiently large number of realisations.
2. From the HEC-DSS result file of the Uncertainty analysis extract and store parameter values for each realisation (using Jython scripts).
3. From the HEC-DSS result file of the Uncertainty analysis and each parameter value extract and store the corresponding outflow result (using Jython scripts).
4. With the stored data records develop the Part 1 – Part 3 GUI of the demo application.

6.1.2. Approach 2: Using multiple runs of HEC-HMS with sampled parameters

As an alternative to Approach 1, the participants may decide to code the necessary functionalities for: generating different parameter values, running HEC-HMS for each parameter value and extracting and storing the outflow results for each parameter value. This would be an operation similar to the one depicted in Figure 4. The sampling of parameters may be random from a particular probability distribution, or ordered from the specified range. It is important to note that the variation of the parameter values for each model run needs to be carried out via updating the input file `*.basin` (see Figure 3). For easier data manipulation, for each model run the file name of the `*.basin` file may be changed, as well as other file names for the control or run specification of HEC-HMS, or the resulting `*.dss` file. The actual model run can be via a batch file, as explained on page 6 of this document. Finally, the outflow extraction from the corresponding `*.dss` file can be carried out using the Jython scripts.

In summary the steps for this approach may be as follows:

1. For each model parameter generate sufficiently large number of values within the specified range (random or ordered).
2. For each parameter value prepare an updated input file `*.basin`, where the new parameter value will be specified. Rename input files as necessary.
3. Run HEC-HMS model as a batch file.
4. From the HEC-DSS result file of the model run extract and store the outflow result and associate it with the corresponding parameter value (using Jython scripts).
5. With the stored data records develop the Part 1 – Part 3 GUI of the demo application.

This approach is somewhat more elaborate than Approach 1, but the developers may have more control over the process of data generation.

6.2. Guidelines for part 2 – part 3 of the demo application: sensitivity analysis of precipitation inputs

Unfortunately the uncertainty analysis of HEC-HMS is not designed for testing uncertainties with respect to precipitation inputs. This means that Approach 1 from the previous sub-section is not available for the sensitivity analysis of the precipitation inputs. Therefore an approach which is very similar to Approach 2 of the previous sub-section needs to be followed, in which HEC-HMS model needs to be run with different precipitation inputs.

It needs to be noted that when precipitation inputs are to be varied outside of the GUI of HEC-HMS this is best achieved via a HEC-DSS file. During the training exercises the precipitation inputs for the Siron model were provided as ‘Manual Entry’. After the model run this input (that has been entered manually) has in fact been stored in a `*.dss` file. Therefore this input can be changed to an input from a DSS file from the GUI of HEC-HMS (e.g. using ‘Multiple Record HEC-DSS’ as ‘Data source’ for the time series of the Precipitation Gauge). This file can then be accessed to vary the precipitation input within a specified range.

The summary steps for this part would be as follows:

1. Select certain range within which the precipitation input will be varied (e.g. ± 30%).
2. Generate sufficiently large number of precipitation coefficients, random or ordered) within the specified range. (With these coefficients the base precipitation input will be multiplied)
3. Access the *.dss file where precipitation input is stored and for each precipitation coefficient update the base precipitation input (using Jython scripts). Rename input files, if necessary.
4. Run HEC-HMS model as a batch file.
5. From the HEC-DSS result file of the model run extract and store the outflow result and associate it with the corresponding precipitation coefficient (using Jython scripts).
6. With the stored data records develop the Part 1 – Part 3 GUI of the demo application.

7. Final remarks

This is only a guidance document for the proposed assignment. The participants are free to vary their approaches as they find it appropriate, as long as the final goals are achieved. Nevertheless, during the final presentation and in their final report they will need to explain which approaches they have taken and why.

Two possible alternative approaches are as follows:

- The Jython scripting support within HEC-DSSVue is not in a professional development environment. Some participants may prefer to use Jython in some other professional environments with which they are more familiar (e.g. Eclipse, NetBeans). They may explore how to bring the required HEC libraries (modules) in those environments and carry out their development there.
- For those familiar with Java programming there is a short guidance on developing Java plug-ins for HEC-DSSVue in Appendix B of the HECDSSVue User Manual. This can be explored as an alternative to Jython scripting.

Finally, the participants are free to extend the assignment in other directions if they have time and interest. Some possibilities are:

- Developing the demo application as a web-based application
- Deploying the model runs and data preparation components in a cloud environment as web services
- Extending the capabilities of the demo application, e.g, for testing joint influence of two parameters, or of parameters and inputs.

Most relevant resources

HEC-HMS User manual, available in the ‘doc’ sub-folder of the HEC-HMS installation folder, and at:

http://www.hec.usace.army.mil/software/hec-hms/documentation.aspx

HEC-DSS User Manual, available in the ‘users-manual’ sub-folder of the HEC-DSSVue installation folder, and at:

http://www.hec.usace.army.mil/software/hec-dssvue/documentation.aspx

Jython web site:

http://www.jython.org/