GPS-X/Python/Matlab Link

You can interact with the Matlab Engine from GPS-X's Python Script Manager. It is assumed that you have Python v2.7 or higher installed (GPS-X comes with Python v2.7 and v3.7) and that you are licensed for, and know how to use, the Python Script Manager in GPS-X (see chapter 14 of the GPS-X User Guide). You will need Matlab v2014b (64 bit) or higher installed. See [MATLAB Products Python Compatibility (mathworks.com)]

The first step is to install Matlab's Python package. Here it is assumed that you have selected GPS-X's default Python 3.7 distribution.

Open a Windows Command Prompt and execute the following (you might need administrator privileges to execute these commands):

- cd "*matlabroot*\extern\engines\python"
- "*gpsxinstalldir*\python37\python.exe" setup.py install

Replace *matlabroot* and *gpsxinstalldir* with actual locations. If you are configured to use a different Python distribution then change the path in the second step to point to that python.exe

Example Usage

- Start GPS-X and Select File>Open…
- Browse to:
  "*gpsxinstalldir*\acm\demo\noise\" and load noise.lyt.
- Switch to Simulation Mode.
- Create a new scenario named "python"
- Right click on the layout background and select System>Input Parameters>Simulation Tool Settings. Scroll down and turn off "Matlab link control"
- Select Tools>Python Script Manager. Click New to create a python script file. Then Click Edit and make the following changes to the script.

```
import matlab.engine
import numpy as np

try:
 eng
except NameError:
 eng = None

if eng == None:
 eng = matlab.engine.start_matlab()
 sd = eng.clock()
 sdnp = np.array(sd._data)
 sum = 0
 for x in sdnp:
   sum = sum + 100 * x
 eng.rng(int(sum))

qinf_sp = gpsx.getValue('qinf_sp')
```

```
incr = 0.0

# start() function executed once at simulation start
#
def start():
 try:
   qconinf = qinf_sp
   gpsx.setValue('qconinf', qconinf)
 except Exception as e:
   print(e)

# cint() function executed at every communication interval
#
def cint():
 global incr
 try:
   incr = incr + 10.0
   qconinf = qinf_sp + 100.0 * eng.randn(1,1)
   gpsx.setValue('qconinf', qconinf)
 except Exception as e:
   print(e)

# eor() function executed once at end of simulation
# finished set True is required to terminate the runSim() function
#
def eor():
 global finished
 finished = True
 try:
   pass
 except Exception as e:
   print(e)

# runSim() call starts simulation in GPS-X
try:
 runSim()
except Exception as e:
 print(e)
```

When you run the simulation from the Python Script Manager, you should see a random signal added to the influent flow.