Transient Simulation in water Networks (TSNet) **WDSA CCWI July 18, 2022**

Tutorial 1: Short intro to TSNet

Instructors: Lu Xing, Lina Sela

In this tutorial, we will introduce the basic functionalities of the TSNet Python package. Detailed information can be found in the online documentation `https://tsnet.readthedocs.io/`.

This tutorial provides the commands and procedures used in the short course organized during the *2nd international Joint Conference on Water Distribution Systems Analysis & Computing and Control in the Water Industry (WDSA & CCWI)* in Valencia, July 18, 2022, `https://wdsa-ccwi2022.upv.es/transient-simulations-in-water-networks/`.

For any questions, please email us:

Lu Xing lu.xing@xylem.com
Lina Sela linasela@utexas.edu

Contents

1.1 Installation

TSNet was tested with Python versions 3.5, 3.6, and 3.7. It can be installed on Windows, Linux, and Mac OS X operating systems. Python distributions, such as Anaconda, are recommended to manage the Python environment as they already contain (or easily support installation of) many Python packages (e.g., SciPy, NumPy, pandas, pip, matplotlib, etc.) that are used in the TSNet package. For more information on Python package dependencies, see online documentation.

1.1.1 Setting up Python Environment

1. Download Anaconda from here.
2. Open your terminal (mac) or command prompt (windows).
3. We recommend creating a new environment to make sure the installation will not mess up with your existing environment. Make sure to specify Python version 3.7 (TSNet will not work with more recent versions).
4. To create a new conda environment type:
   ```
   conda create --name tsnet python=3.7
   ```
5. Activate the newly created environment
   ```
   conda activate tsnet
   ```
6. Install tsnet
   ```
   pip install tsnet
   ```
7. Check the installation
   ```
   pip show tsnet
   ```

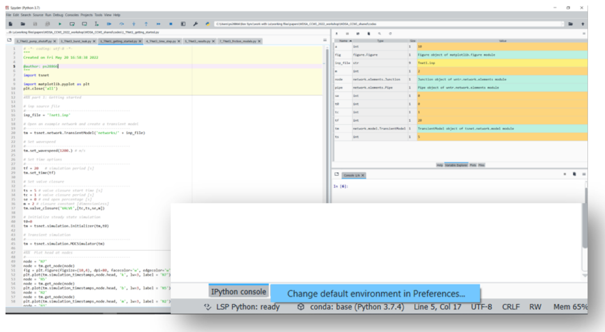
1.1.2 Getting Spyder and Jupyter working with the new environment

After creating a new environment, you will need to get Spyder and Jupyter Notebooks to work with the new environment. There are several ways to do that, below are some suggestions, and more information can be found in Spyder documentation and Towards data science.

1.1.2.1 Getting Spyder: The Easy Way

1. After creating the new environment, open Spyder in your base environment.
2. Click the name of the environment in the status bar (bottom right).
3. Click `'Change default environment in Preferences'` (see figure below).
4. Select `'Use the following Python interpreter'` (see figure below).

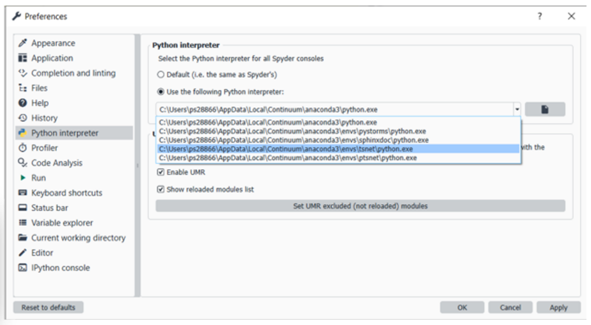
5. Select your environment from the drop-down list and click `'OK'`.
6. You should see `tsnet` environment in the status bar.



```
# -*- coding: utf-8 -*-
"""
Created on Fri May 20 16:58:38 2022

@author: ps28866
"""
import tsnet

import matplotlib.pyplot as plt
plt.close('all')

#%% part 1: Getting started

# inp source file
#-------
inp_file = 'Tnet1.inp'

# Open an example network and create a transient model
#-------
tm = tsnet.network.TransientModel('networks/' + inp_file)

# Set wavespeed
#-------
tm.set_wavespeed(1200.) # m/s

# Set time options
#-------
tf = 20   # simulation period [s]
tm.set_time(tf)

# Set valve closure
#-------
ts = 5 # valve closure start time [s]
tc = 1 # valve closure period [s]
se = 0 # end open percentage [s]
m = 2 # closure constant [dimensionless]
tm.valve_closure('VALVE',[tc,ts,se,m])

# Initialize steady state simulation
t0=0
tm = tsnet.simulation.Initializer(tm,t0)

# Transient simulation
#-------
tm = tsnet.simulation.MOCSimulator(tm)

#%% Plot head at nodes
#-------
node = 'N7'
node = tm.get_node(node)
fig = plt.figure(figsize=(10,4), dpi=80, facecolor='w', edgecolor='w'
plt.plot(tm.simulation_timestamps,node.head, 'k', lw=3, label = 'N7')
node = 'N5'
node = tm.get_node(node)
plt.plot(tm.simulation_timestamps,node.head, 'b', lw=3, label = 'N5')
node = 'N2'
node = tm.get_node(node)
plt.plot(tm.simulation_timestamps,node.head, 'm', lw=3, label = 'N2')
```

Name	Type	Size	Value
a	int	1	10
fig	figure.Figure	1	Figure object of matplotlib.figure module
inp_file	str	9	Tnet1.inp
m	int	1	2
node	network.elements.Junction	1	Junction object of wntr.network.elements module
pipe	network.elements.Pipe	1	Pipe object of wntr.network.elements module
se	int	1	0
t0	int	1	0
tc	int	1	1
tf	int	1	20
tm	network.model.TransientModel	1	TransientModel object of tsnet.network.model module
ts	int	1	5

```
In [6]:
```



Python interpreter

Select the Python interpreter for all Spyder consoles

○ Default (i.e. the same as Spyder's)

● Use the following Python interpreter:

C:\Users\ps28866\AppData\Local\Continuum\anaconda3\python.exe

```
C:\Users\ps28866\AppData\Local\Continuum\anaconda3\python.exe
C:\Users\ps28866\AppData\Local\Continuum\anaconda3\envs\pystorms\python.exe
C:\Users\ps28866\AppData\Local\Continuum\anaconda3\envs\sphinxdoc\python.exe
C:\Users\ps28866\AppData\Local\Continuum\anaconda3\envs\tsnet\python.exe
C:\Users\ps28866\AppData\Local\Continuum\anaconda3\envs\ptsnet\python.exe
```

☑ Enable UMR

☑ Show reloaded modules list

Set UMR excluded (not reloaded) modules

1.1.2.2 Getting Jupyter Notebooks: The Easy Way

1. Open Jupyter Notebooks from your base environment.
2. Your kernel list will only show your current environment (see figure below). To get your other environment kernels to show:
3. Install `nb_conda_kernels` in your base environment:

   ```
   conda install nb_conda_kernels
   ```

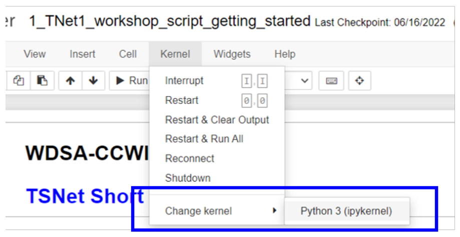
4. Activate the environment you want to use in your notebook

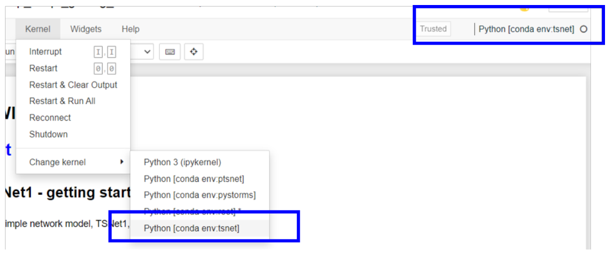
   ```
   conda activate tsnet
   ```

5. Install `iypkernel`:

   ```
   conda install ipykernel
   ```

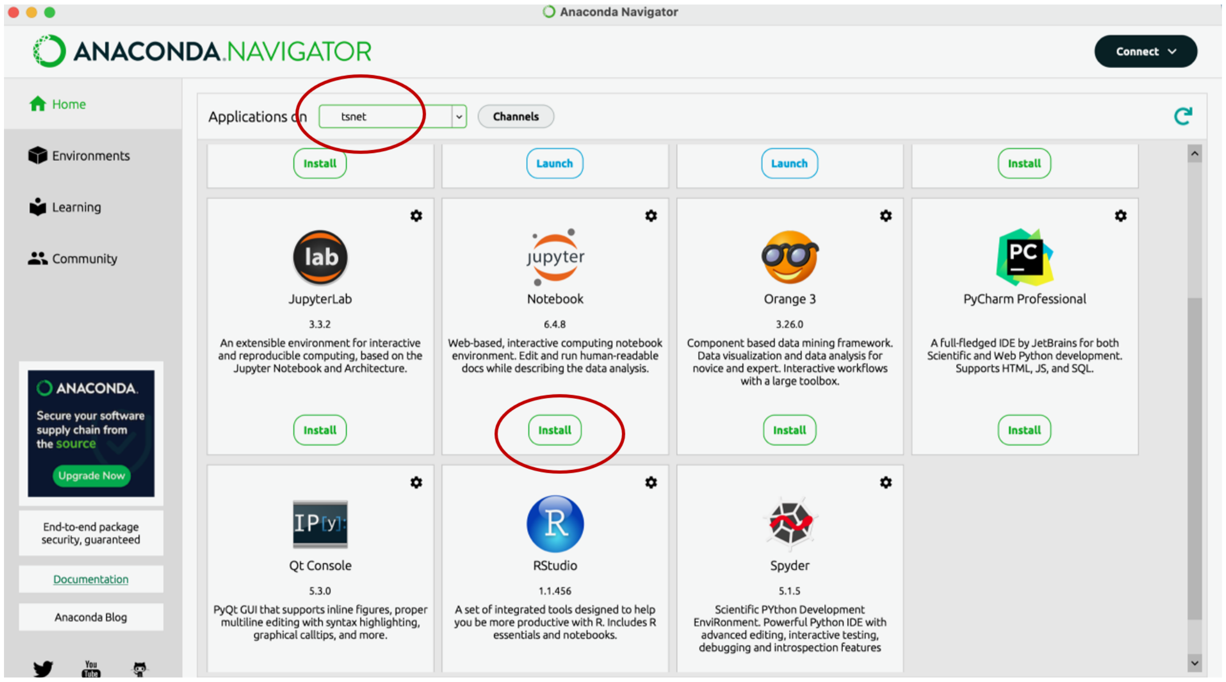
6. Restart Jupyter Notebooks from your base environment (you might need to restart Anaconda Navigator as well)
7. Your environments should appear now in your kernel list and you can switch between them (see figure below).





1.1.2.3 The Quick and Dirty Way

Install via the Anaconda Navigator or terminal, while in the `tsnet` environment. See the figure below for example.



An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox.

A full-fledged IDE by JetBrains for both Scientific and Web Python development. Supports HTML, JS, and SQL.

PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.

Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

1.1.3 Troubleshooting

Common issues include:

- *Python version:* TSNet works with Python 3.5, 3.6, and 3.7. If you have a more recent Python version you will get an error when trying to run TSNet models. To solve this issue, check your Python version and if you have have a more recent one, use the following command when creating the tsnet environment:

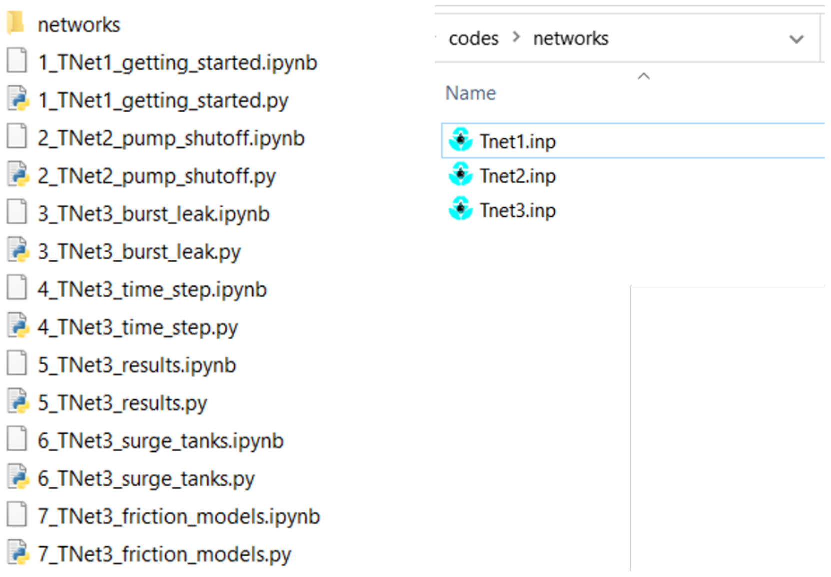
  ```
  conda create --name tsnet python=3.7
  ```

- *WNTR version:* TSNet works with WNTR 0.2.3. Some functionalities of TSNet might not work with more recent versions. Check your WNTR version, and if needed install version 0.2.3 using (you can do this in Spyder, make sure you are in the `tsnet` environment).

  ```
  pip install wntr==0.2.3
  ```

1.2 Scenarios

The following subsections include codes for setting up and running different transient models. All the source codes and three test networks are available here `https://utexas.box.com/s/io2azj1bci4gtdrouz3xrd629tb6k1wi` in `.py` and `.ipynb` formats that can be executed using Spyder or Jupyter Notebooks (see figure below). For each scenario, we briefly summarize the main points that will be covered during the course.



networks
1_TNet1_getting_started.ipynb
1_TNet1_getting_started.py
2_TNet2_pump_shutoff.ipynb
2_TNet2_pump_shutoff.py
3_TNet3_burst_leak.ipynb
3_TNet3_burst_leak.py
4_TNet3_time_step.ipynb
4_TNet3_time_step.py
5_TNet3_results.ipynb
5_TNet3_results.py
6_TNet3_surge_tanks.ipynb
6_TNet3_surge_tanks.py
7_TNet3_friction_models.ipynb
7_TNet3_friction_models.py

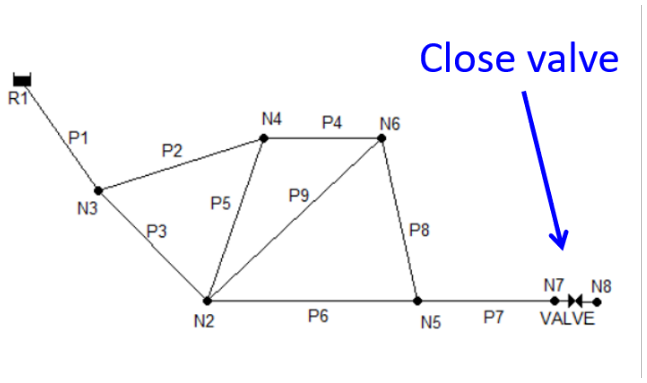
codes > networks

Name
Tnet1.inp
Tnet2.inp
Tnet3.inp

1.2.1 Scenario 1 - Getting started

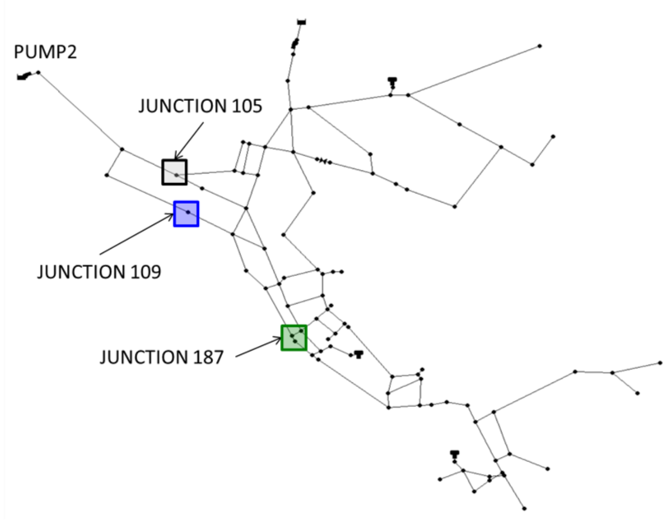
We will start with a simple network model, Tnet1, to demonstrate how to:

1. Import tsnet
2. Generate a transient model
3. Set wave speed
4. Set time step and simulation period
5. Perform initial condition calculation
6. Create transient event by closing a valve
7. Run transient simulation
8. Extract and plot simulation results



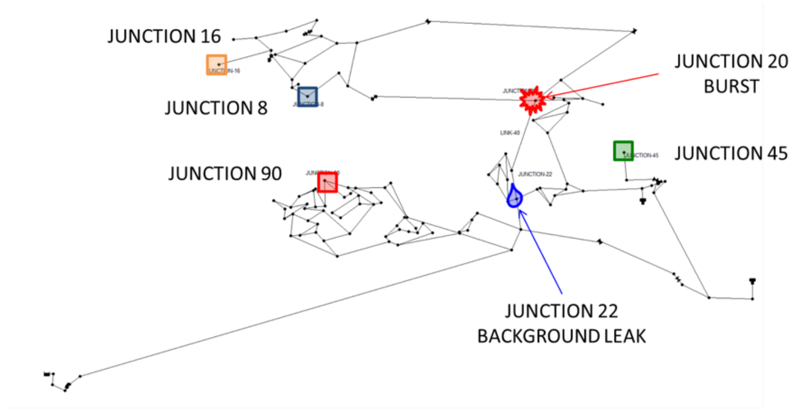
1.2.2 Scenario 2 - Pump shutoff

In this example, we will use Tnet2 to demonstrate how to simulate pump shut-off and extract results. We will also demonstrate how to (1) specify demand- or pressure-driven simulation in the calculation of initial conditions, and (2) how to save simulation results.



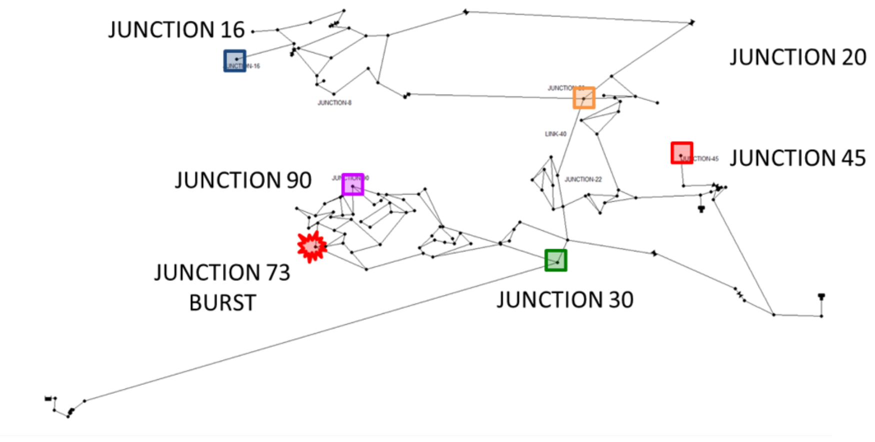
1.2.3 Scenario 3 - Burst event with a background leak

In this example, we will use Tnet3 to demonstrate how to simulate burst event with a background leak. We will: (1) generate a burst at Junction-20, (2) add an existing leak at Junction-22, (3) plot leak and burst discharges, and (4) plot head response in different locations in the network.



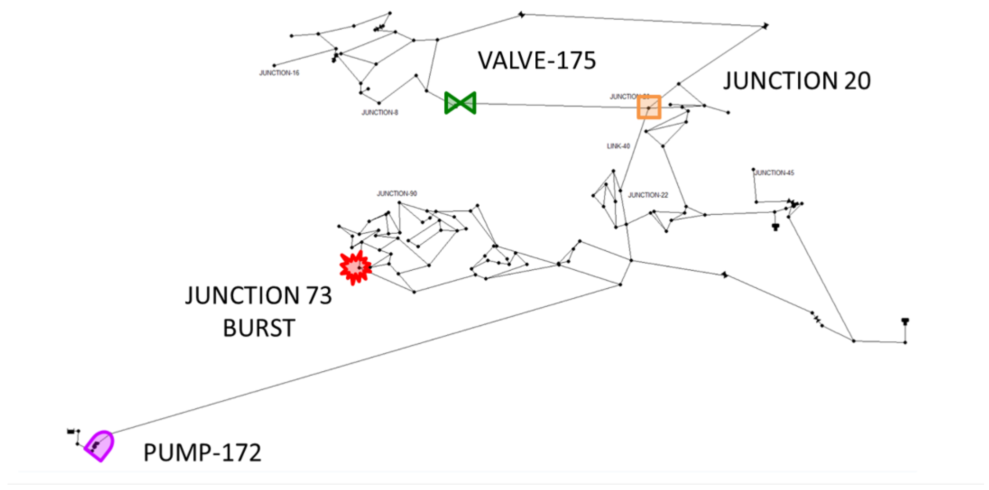
1.2.4 Scenario 4 - Selecting time step

In this example, using Tnet3, we will demonstrate the importance of selecting the time step for the simulation and demonstrate how it affects the results and running times. We will discuss how TSNet determines the time step and how a user can specify custom time step. More information, including detailed examples can be found in the online documentation.



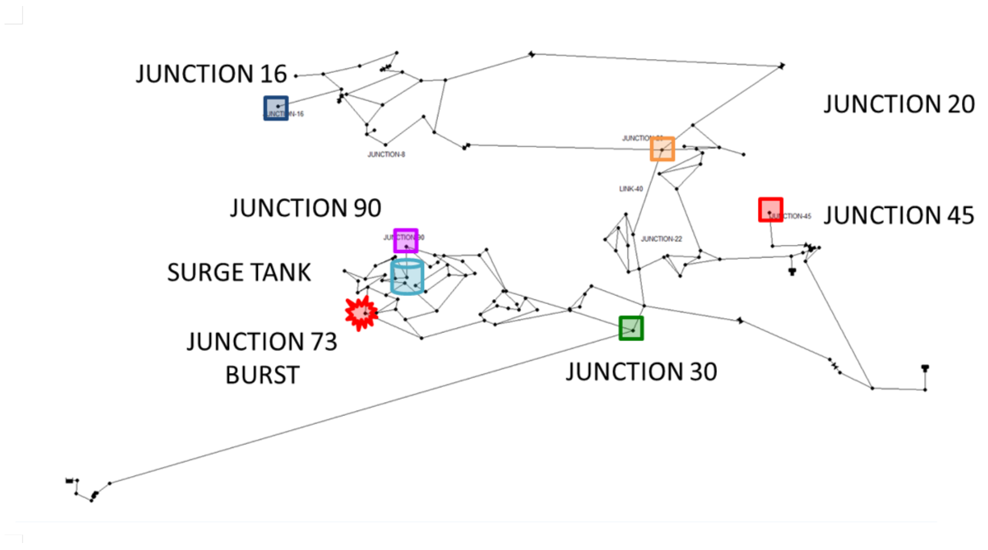
1.2.5 Scenario 5 - Working with results

In this example, we will demonstrate how to work with results from previous transient simulations using the `pickle` object. We will load simulation results from transient events generated in TNet3 by closing a pump, closing a valve, and a burst, and plot and compare the transient pressure in Junction-20.



1.2.6 Scenario 6 - Adding surge tanks

In this example, we will generate a burst in Junction-73 and compare the transient response in different locations in the network in the following cases: (1) without a surge tank, (2) with open surge tank, and (3) with closed surge tank.



1.2.7 Scenario 7 - Testing friction models

In this example, we will again generate a burst in Junction-73 (as in Scenario 4) and compare simulation results and computational times with different friction models: (1) steady, (2) quasi-steady, and (3) unsteady.