Tellurium and libRoadRunner in a Nutshell

Herbert M Sauro

University of Washington

hsauro@uw.edu

Developed by Andy Somogyi and the software team at UW with input from Maciek Swat

July 26, 2020

Tellurium

Tellurium is an integrated platform for Python users. I can be use with IDEs such as Jupyter, Spyder, etc. It runs It runs on Mac, Windows and Linux. When distributed with Windows it comes with the latest spyder IDE 4.1.3 and Juypter nodebook IDE. It includes the following main libraries:

libRoadRunner: A high performance SBML simulation library.

Antimony: Allows user to write models in a more human readable form.

SBML2Matlab: Allows users to export models in Matlab format

libsbml: Allows users to manipulate SBML

Other libraries include libCombine, and libSEDML. In addition Tellurium comes preloaded with the Python plotting library **Matplotlib**, the array package **numpy** and **scipy** as well as a large number of other packages. . Tellurium also comes with a number of helper subroutines to make it easier for the average modeler including a very easy to use model fitting pacagae.

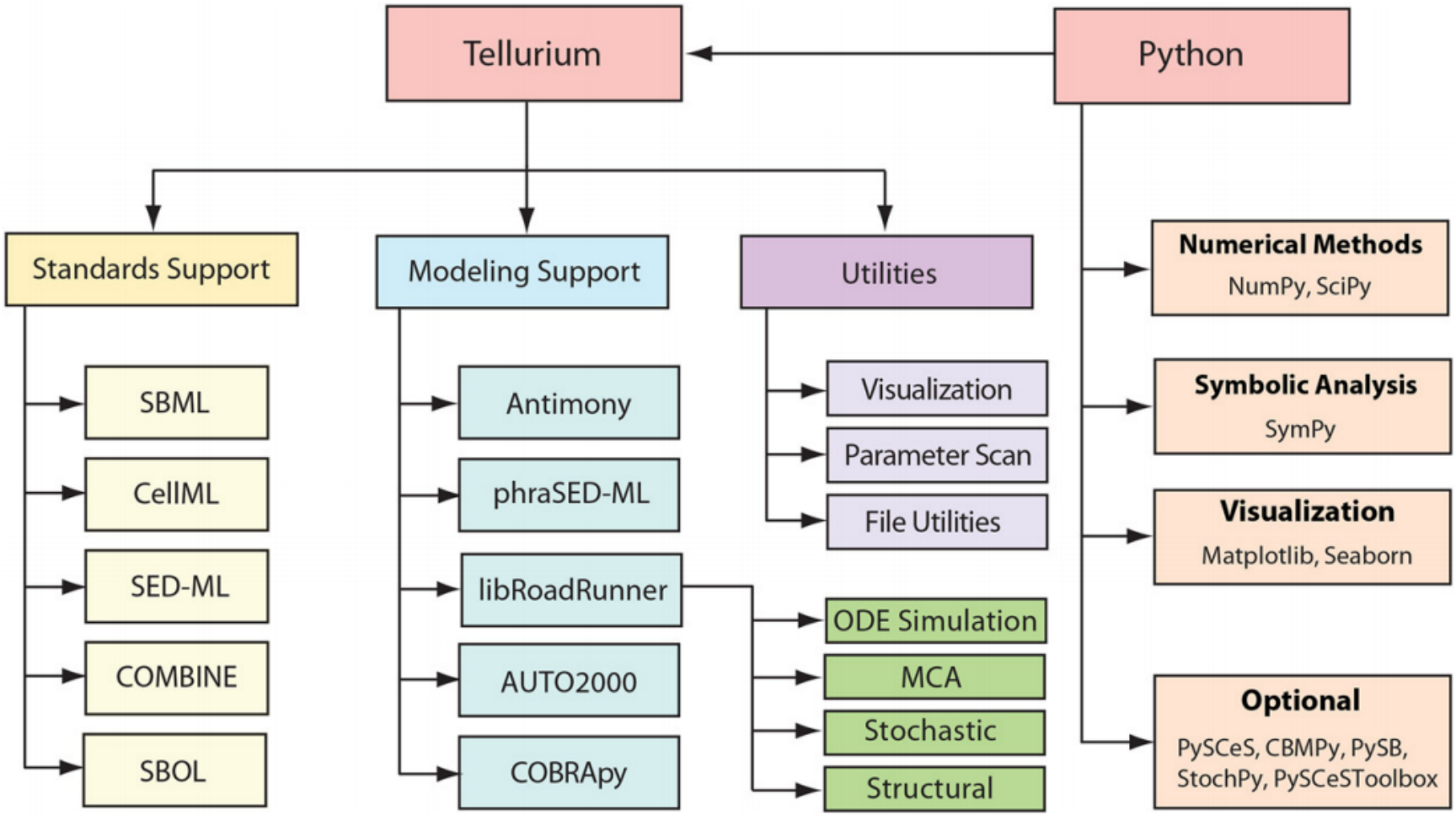
Download the Software

For the workshop we will be using nanohub but you can also run Tellurium on your desktop or laptop machine. To download the software go to the web site:

`http://tellurium.analogmachine.org/`

Simply pick the download that is appropriate for your computer's operating system.

High Level View



First Example

Example

```
import tellurium as te

r = te.loada ('''
    S1 -> S2; k1*S1;
    S2 -> S3; k2*S2;

    k1 = 0.1; k2 = 0.45;
    S1 = 10; S2 = 0; S3 = 0
''')

result = r.simulate (0, 40, 100)
r.plot ()
```

Antimony

Example (Simple Model)

```
S1 -> S2; k1*S1;

k1 = 0.1; S1 = 10; S2 = 0
```

$$\frac{dS_1}{dt} = -k_1 S_1$$

$$\frac{dS_2}{dt} = k_1 S_1$$

Antimony

Example (Multiple Reactions)

```
S1 -> S2; k1*S1;
S2 -> S3; k2*S2;

k1 = 0.1; k2 = 0.2;
S1 = 10; S2 = 0; S3 = 0
```

$$\frac{dS_1}{dt} = -k_1 S_1$$

$$\frac{dS_2}{dt} = k_1 S_1 - k_2 S_2$$

$$\frac{dS_3}{dt} = k_2 S_2$$

Antimony

Example (Rate Laws)

```
S1 -> S2; k1*S1 - k2*S2;      # Reversible
S2 -> S3; Vmax*S3/(Km + S3); # Michaelis-Menten

k1 = 0.1; k2 = 0.2; Vmax = 10; Km = 0,4
S1 = 10; S2 = 0; S3 = 0
```

Antimony

Example (Bimolecular Reactions)

```
S1 + S2 -> S3; k1*S1*S2;
S3 -> S4 + S4; k2*S3;

k1 = 0.1; k2 = 0.2;
S1 = 10; S2 = 0; S3 = 0
```

Antimony

Example (Fixed Species)

```
# This is a comment
# A $ means FIX the concentration of the species
$S1 -> S2; k1*S1;
S2 -> $S3; k2*S2;

k1 = 0.1; k2 = 0.2;
S1 = 10; S2 = 0; S3 = 0
```

Example (Events)

```
# This is a comment
# A $ means FIX the concentration of the species
$S1 -> S2; k1*S1;
S2 -> $S3; k2*S2;

at (time > 5): k2 = k2*2;

k1 = 0.1; k2 = 0.2;
S1 = 10; S2 = 0; S3 = 0
```

Antimony

Example (Named Reactions)

```
# Name reactions are useful for getting the reaction rates
J1: $S1 -> S2; k1*S1;
J2: S2 -> $S3; k2*S2;

k1 = 0.1; k2 = 0.2;
S1 = 10; S2 = 0; S3 = 0
```

Antimony

Example (Loading a Model into libRoadRunner)

```
import tellurium as te
r = te.loada ('''
  J1: $S1 -> S2; k1*S1;
  J2: S2 -> $S3; k2*S2;

  k1 = 0.1; k2 = 0.2;
  S1 = 10; S2 = 0; S3 = 0
''')
```

Antimony

Example (Standard import boiler plate)

```
import tellurium as te
import numpy
import roadrunner
import matplotlib.pyplot as plt
```

Antimony

Example (Run a Simulation)

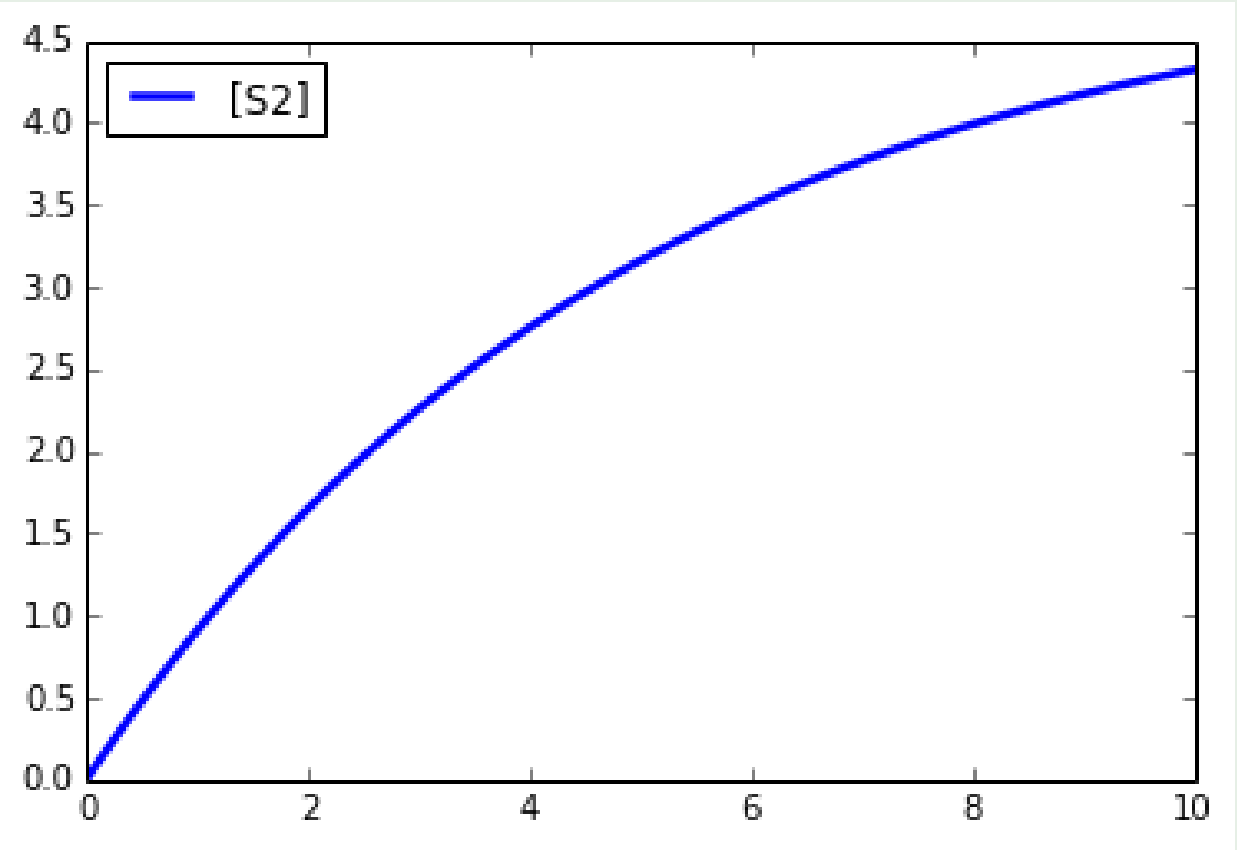
```
r = te.loada ('''
  J1: $S1 -> S2; k1*S1;
  J2: S2 -> $S3; k2*S2;

  k1 = 0.1; k2 = 0.2;
  S1 = 10; S2 = 0; S3 = 0
''')

result = r.simulate (0, 10, 100)
```

Antimony

Example (Plotting Results)

```
r = te.loada ('''
  J1: $S1 -> S2; k1*S1;
  J2: S2 -> $S3; k2*S2;

  k1 = 0.1; k2 = 0.2;
  S1 = 10; S2 = 0; S3 = 0
''')

result = r.simulate (0, 10, 100)
r.plot ()
```

Example (Plotting Results)



Antimony

Example (Changing Values)

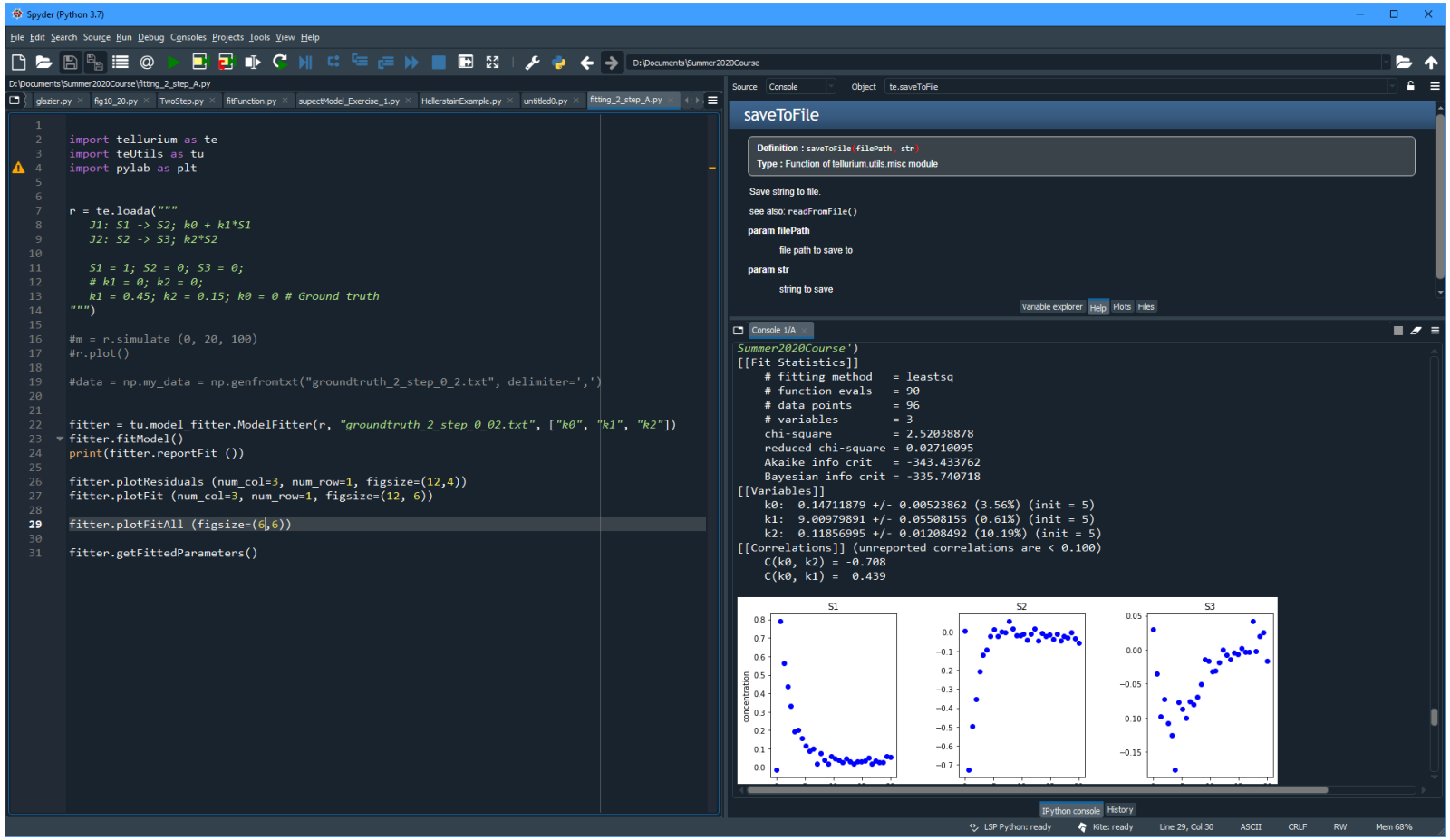
```
r = te.loada ('''
  J1: $S1 -> S2; k1*S1;
  J2: S2 -> $S3; k2*S2;

  k1 = 0.1; k2 = 0.2;
  S1 = 10; S2 = 0; S3 = 0
''')

r.k1 = 12.3
r.S1 = 20
result = r.simulate (0, 10, 100)
r.plot ()
```

Example (Resetting the Model)

```
r = te.loada ('''
  J1: $S1 -> S2; k1*S1;
  J2: S2 -> $S3; k2*S2;

  k1 = 0.1; k2 = 0.2;
  S1 = 10; S2 = 0; S3 = 0
''')

result = r.simulate (0, 10, 100)

r.reset()  # Reset to species initial conditions
r.resetAll() # Reset initial conditions and parameter values
r.resetToOrigin() # Reset back to when the model was loaded
```

Demo

Tellurium Interface when using Spyder



Documentation

Go to:

tellurium.analogmachine.org

and

libroadrunner.org

and select the documentation menu. Note that the tellurium documentation also includes roadrunner.

Exercise

Build a model that describes two consecutive reactions, each reaction governed by the simple Michaelis-Menten rate law

$$v = V_m \frac{S}{K_m + S}$$

$$\$S_1 \rightarrow S_2 \rightarrow \$S_3$$

Note S_1 and S_3 are FIXED. Set the parameters and species to:

```
Km1 = 0.5; Km2 = 0.5;
S1 = 10; S2 = 0; S3 = 0;
Vm1 = 30; Vm2 = 20;
```

Load the model into libroadrunner and run a simulation from time zero to time 10 time units. Plot the results. Explain what you observe. Set Vm1 = 18 and rerun the simulation, explain the results.