

Tellurium and libRoadRunner in a Nutshell

Herbert M Sauro

University of Washington

hsauro@uw.washington.edu

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

August 10, 2014

Tellurium is an integrated platform based on Python and spyder2. It runs on Mac, Windows and Linux. It includes the following 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

In addition Tellurium comes preloaded with the Python plotting library **Matplotlib** and the array package **numpy**. Tellurium also comes with a small number of helper subroutines to make it easier for the average modeler.

Download the Software

To download the software go to the web site:

<http://tellurium.analogmachine.org/>

Pick the download that is appropriate for your computer.

Do this now.....

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 (result)
```

Example (Simple Model)

S1 -> S2; k1*S1;

k1 = 0.1; S1 = 10; S2 = 0

$$\frac{dS1}{dt} = -k_1 S_1$$

$$\frac{dS2}{dt} = k_1 S_1$$

Example (Multiple Reactions)

$S_1 \rightarrow S_2; k_1 \cdot S_1;$

$S_2 \rightarrow S_3; k_2 \cdot S_2;$

$k_1 = 0.1; k_2 = 0.2;$

$S_1 = 10; S_2 = 0; S_3 = 0$

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

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

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

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
```

Example (Bimolecular Reactions)

$S1 + S2 \rightarrow S3; k1*S1*S2;$

$S3 \rightarrow S4 + S4; k2*S3;$

$k1 = 0.1; k2 = 0.2;$

$S1 = 10; S2 = 0; S3 = 0$

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
```

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
```

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
''')
```

Example (Standard import boiler plate)

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

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)
```

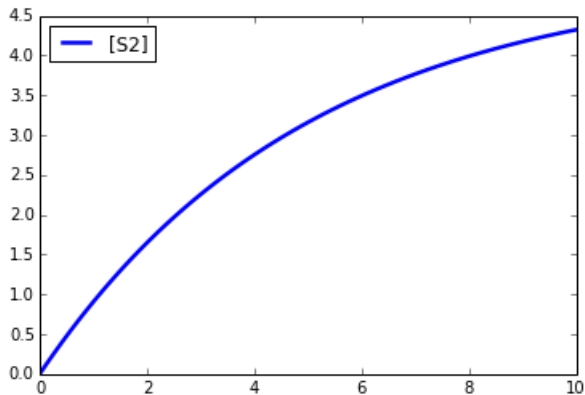
Example (Plotting Results)

```
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 (result)
```

Example (Plotting Results)



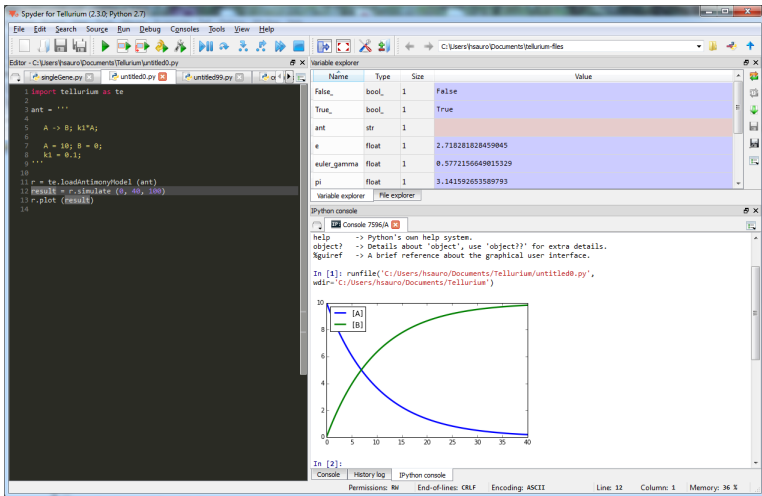
Example (Changing Values)

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

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
```

Tellurium.RoadRunner Interface



Go to:

tellurium.analogmachine.org

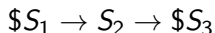
and

libroadrunner.org

Exercise

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

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



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

$$K_{m1} = 0.5; K_{m2} = 0.5;$$

$$S_1 = 10; S_2 = 0; S_3 = 0;$$

$$V_{m1} = 30; V_{m2} = 20;$$

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 $V_{m1} = 18$ and rerun the simulation, explain the results.

The Systems Biology Markup Language (SBML) is a representation format, based on XML, for communicating and storing computational models of biological processes. It is a free and open standard with widespread software support. SBML can represent many different classes of biological phenomena, including metabolic networks, cell signaling pathways, regulatory networks, infectious diseases, and many others. As an XML format, SBML is not meant to be read or written by Humans.