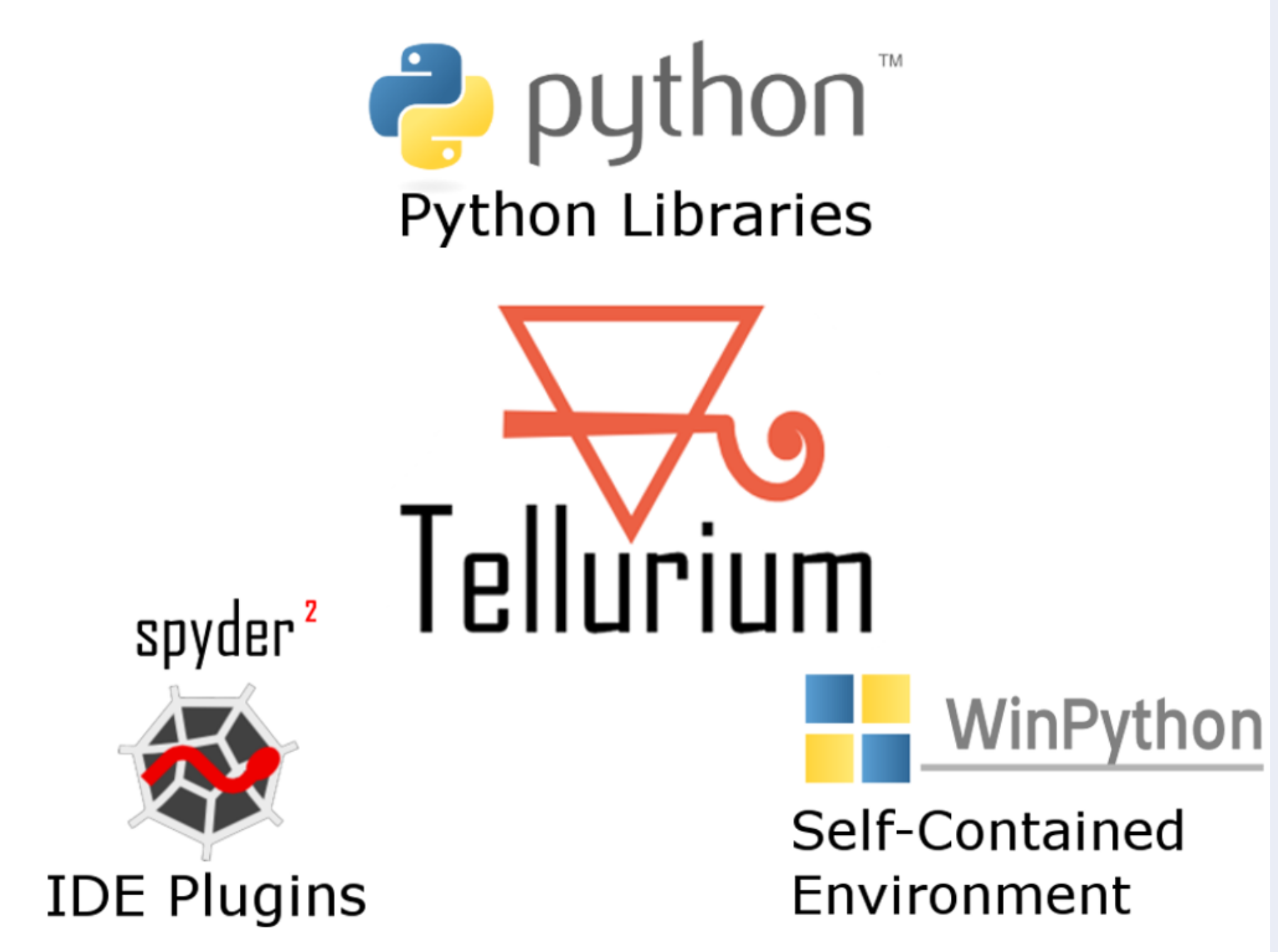
# Tellurium : Fully Integrated Python Tools for Modeling, Analysis and Reproducibility

Kiri Choi[1], Kyle Medley[1], Lucian Smith[1], Stanley Gu[1], Wilbert Copeland[1], and Herbert M Sauro[1]

[1]Department of Bioengineering, University of Washington, Seattle, WA 98195

## Aims

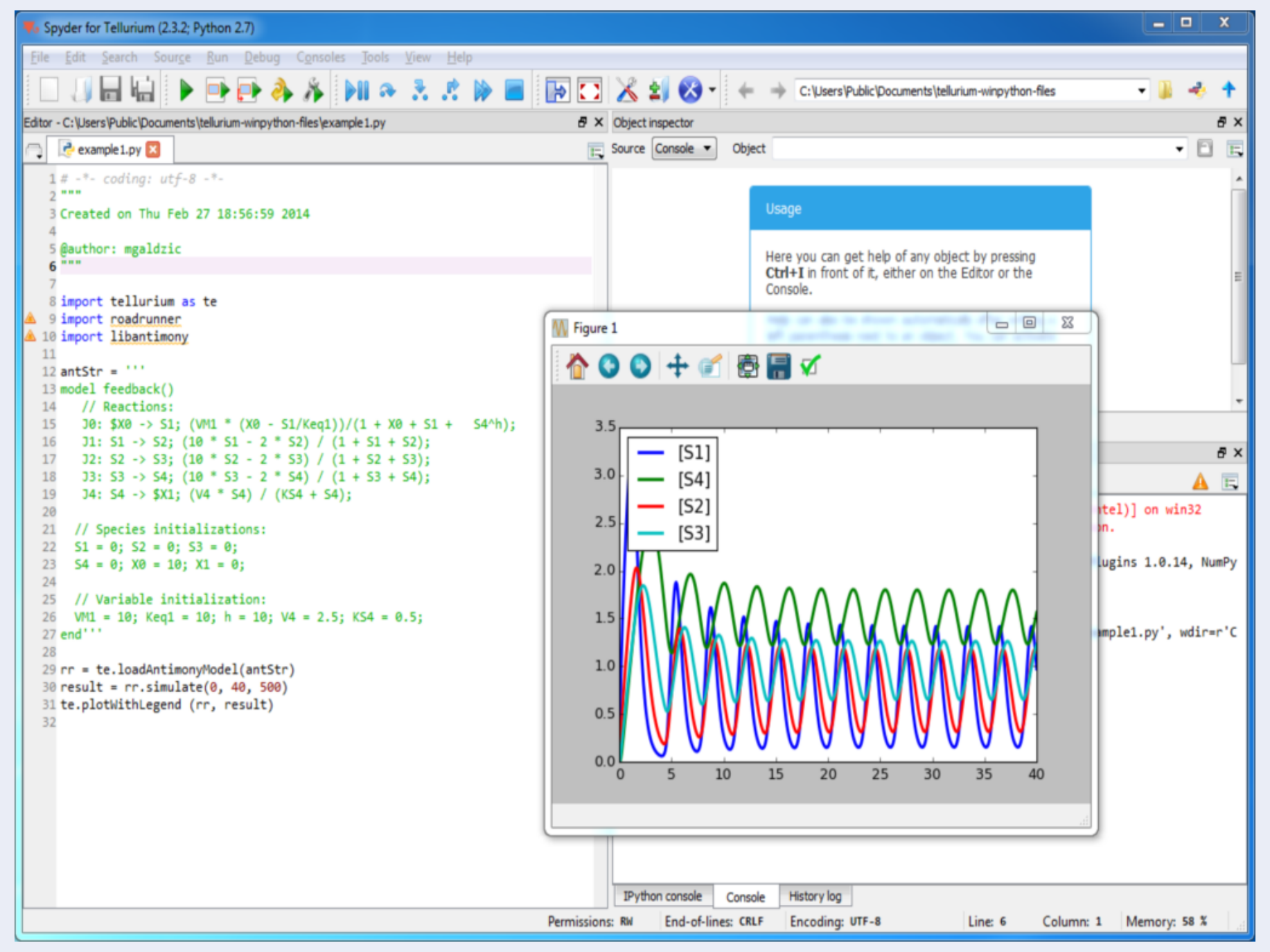
- To develop a powerful and user-friendly modeling platform for systems and synthetic biology
- To develop advanced computing capabilities for the biomedical community
- To develop technologies and standards for model reproducibility
- To support modeling efforts in both academia and industry



## Implementation

Tellurium is a Python based integrated development environment for systems and synthetic biology. Some of the features include:

- Interactive, modular, and flexible platform for modeling
- Fully self-contained and pre-configured environment
- Inclusion of core libraries essential for systems biology simulation
- User-friendly plugins with intuitive UI based on PyQt4
- Customized version of Spyder IDE for high accessibility
- Open platform featuring all the advantages of Python


Spyder for Tellurium (2.3.2; Python 2.7)
File Edit Search Source Run Debug Consoles Tools View Help
C:\Users\Public\Documents\tellurium-winpython-files
Editor - C:\Users\Public\Documents\tellurium-winpython-files\example1.py
example1.py
Object inspector
Source Console
Object
Usage
Figure 1
[S1]
[S4]
[S2]
[S3]
IPython console
Console
History log

```
# -*- coding: utf-8 -*-
"""
Created on Thu Feb 27 18:56:59 2014

@author: mgaldzic
"""

import tellurium as te
import roadrunner
import libantimony

antStr = '''
model feedback()
   // Reactions:
   J0: $X0 -> S1; (VM1 * (X0 - S1/Keq1))/(1 + X0 + S1 +   S4^h);
   J1: S1 -> S2; (10 * S1 - 2 * S2) / (1 + S1 + S2);
   J2: S2 -> S3; (10 * S2 - 2 * S3) / (1 + S2 + S3);
   J3: S3 -> S4; (10 * S3 - 2 * S4) / (1 + S3 + S4);
   J4: S4 -> $X1; (V4 * S4) / (KS4 + S4);

  // Species initializations:
  S1 = 0; S2 = 0; S3 = 0;
  S4 = 0; X0 = 10; X1 = 0;

  // Variable initialization:
  VM1 = 10; Keq1 = 10; h = 10; V4 = 2.5; KS4 = 0.5;
end'''

rr = te.loadAntimonyModel(antStr)
result = rr.simulate(0, 40, 500)
te.plotWithLegend (rr, result)
```

Here you can get help of any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console.

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 6 Column: 1 Memory: 58 %

Figure: Tellurium Main Interface

## Reproducibility

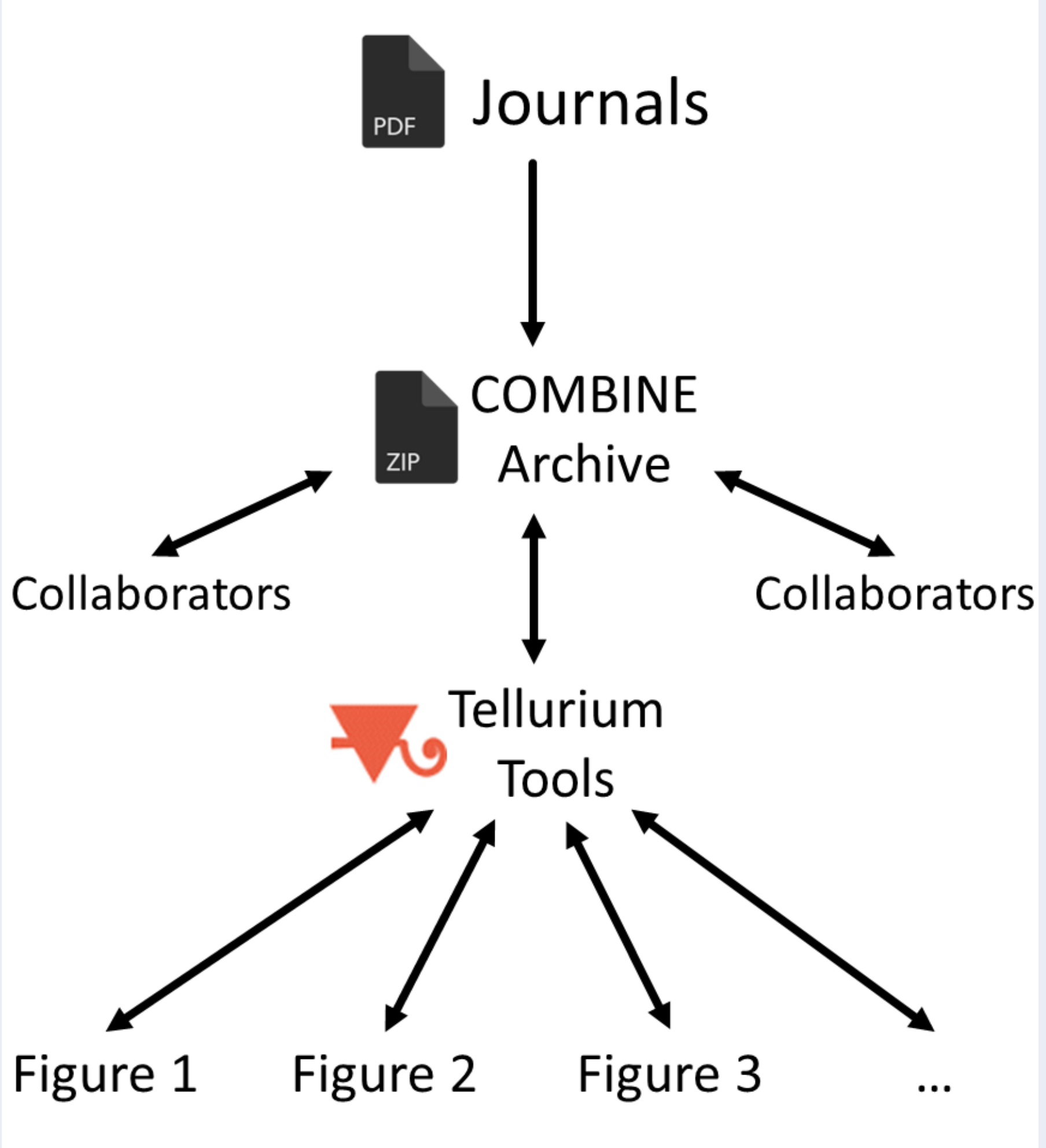
We believe that every computational model should be easily reproducible. That is why we push for standards in systems and synthetic biology community. Our philosophy on reproducibility of computational models is reflected in the way we design our software.

- Tellurium is prepackaged with libSBML, libSEDML, and pySBOL to provide full support for the standards widely adopted in the community.
- Our default simulator fully supports SBML specification.

## Reproducibility

- Tellurium utilizes Antimony and PhrasedML as model and simulation setup definition language. Both languages can be translated to and from SBML and SED-ML files with ease.
- Tellurium comes with wide variety of plugins simplifying the translation process between Python language and SBML/SED-ML files to boost exchangeability.

The basic work flow for operating with exchangeable formats such as the COMBINE archive is presented below:



## Selected List of Libraries

Tellurium comes with number of pre-configured core libraries including but not limited to:

## libRoadRunner

libRoadRunner is a high-performance cross-platform SBML compatible simulation library based on C/C++. In particular, libRoadRunner

- Supports both deterministic and stochastic simulations
- Uses Sundial's CVODE for time course simulations
- Uses NLEQ2 for steady state solutions
- Computes conservation laws to reduce model size
- Computes the frequency response for models
- Comes with integrated bifurcation analysis library

## libRoadRunner

- Utilizes machine code compilation via LLVM
- Fully supports SBML model specification
- Provides full documentations for Python, C++, and C APIs

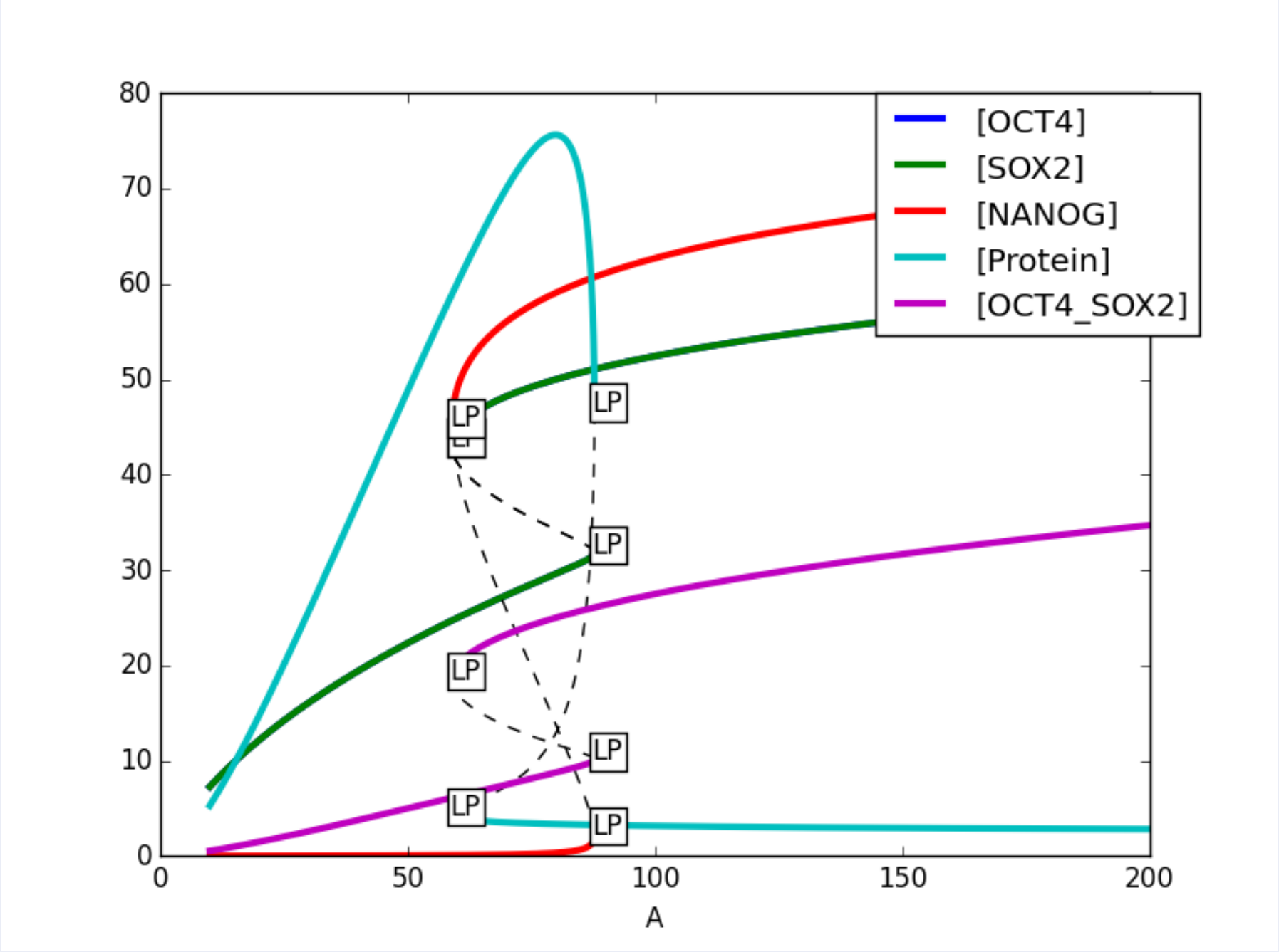


Figure: Example of bifurcation diagram produced in Tellurium (Chickarmane 2006)

### Features Currently under Development

- Data fitting using Levenberg-Marquardt, Nelder-Mead and Evolutionary Algorithms

## Antimony and PhrasedML

Antimony is a human-readable/writable text-based model definition language ideal for defining metabolic, signaling and gene networks. Antimony is modular and is fully compatible with the SBML specification. Similarly, PhrasedML is a text-based simulation experiment definition language designed for better exchangeability between archival formats and programming languages, it is fully compatible with the SED-ML specification.

For example, a simple UniUni reaction can be defined in Antimony as below:

```
S1 -> S2;  k1*S1
S1 = 10;  S2 = 0
k1 = 1
```

Below is a PhrasedML string that performs a uniform time course simulation from time 0 to 10 with 100 points and plots the results.

```
model1 = model "./simpleuniuni.xml"
sim1 = simulate uniform(0, 10, 100)
task1 = run sim1 on model1
plot time vs S1, S2
```

When executed, both Antimony and PhrasedML strings give enough information to plot the graph below:

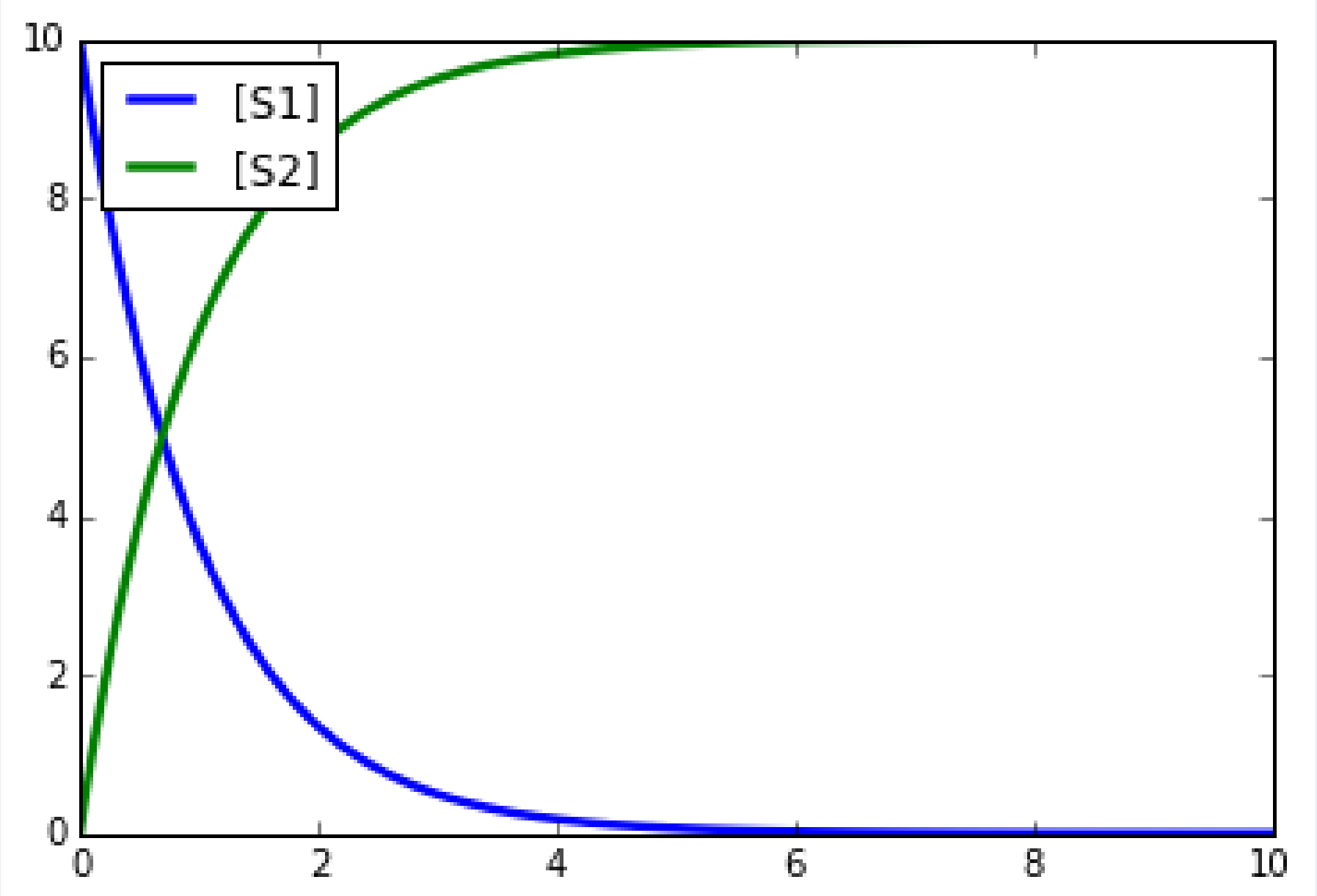


Figure: Output of Antimony and PhrasedML strings

## IDE Plugins

Currently, three major plugins are available for Tellurium. More plugins are planned for the future.

### Import/OpenSBML

Import SED-ML and COMBINE plugins are designed to provide a simple and fast translation from SED-ML files or COMBINE archives to functional and executable Python scripts. We give users options to choose whether to use PhrasedML or basic Python commands when translating SED-ML for increased exchangeability. Utilizing libSBML and libSEDML, Import plugins improve reproducibility by bridging the gap between archival files and executable codes. The OpenSBML plugin accepts valid SBML models and automatically translates the model into an Antimony model without having to perform manual translation va the console

### Network Viewer

The network viewer allows users to quickly visualize network models. The plugin can directly load, modify, and save SBML models without layout information given.

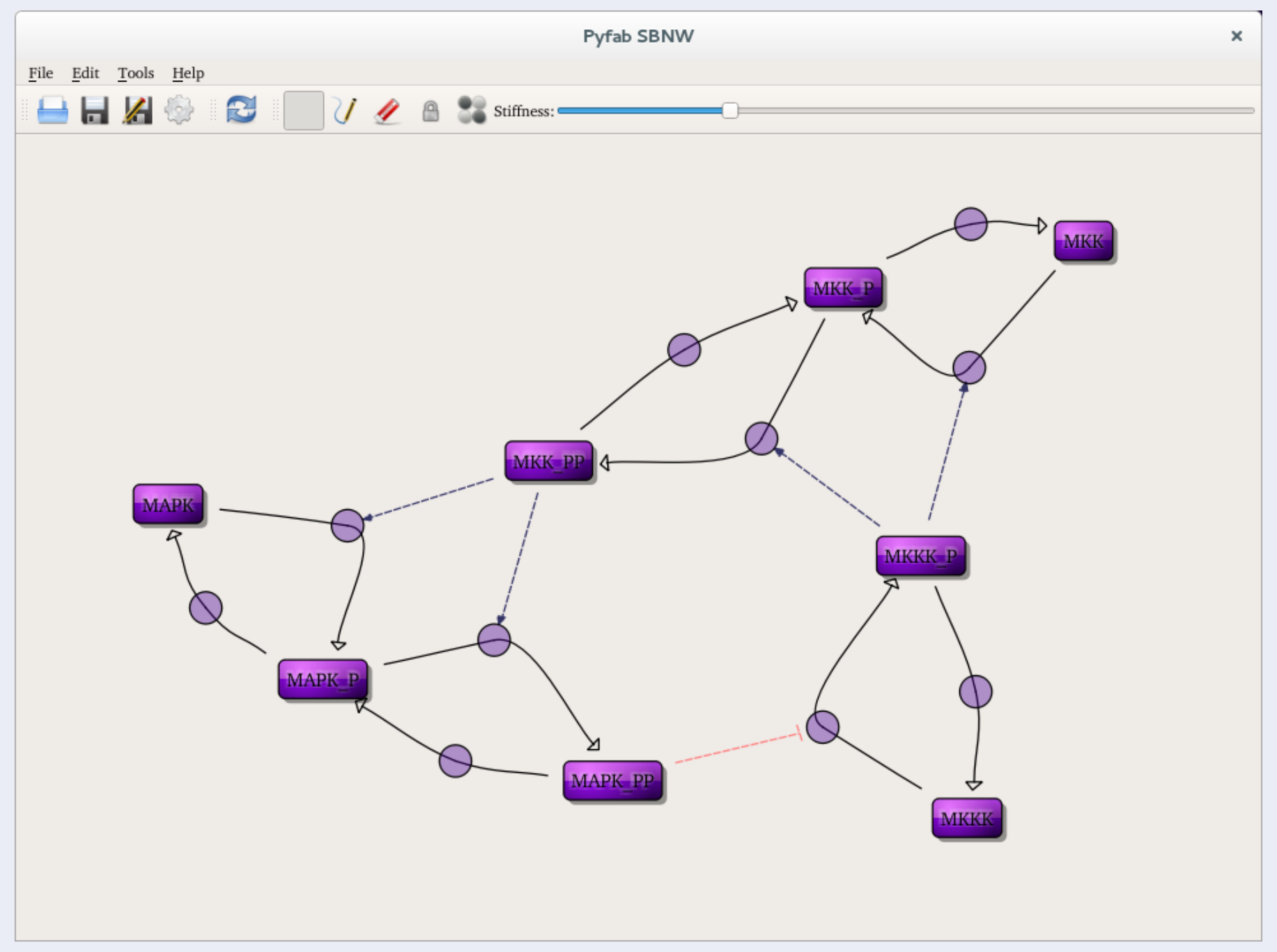


Figure: Network viewer interface (showing MAPK cascade model by Kholodenko, 2000)

### Rate Law Template

Rate Law Template provides a dictionary of equations commonly used in systems and synthetic biology. The plugin provides a simple graphical interface where users can change the parameters and insert the equation directly to the model editor.

## Availability

Installers, source codes, and documentations are available at:

**tellurium.analogmachine.org**

under Apache License 2.0

## Acknowledgements

We thank the following people and organizations for their help, feedback, development expertise, and advice: Frank Bergmann, Totte Karlsson, Andy Somogyi, and Anastasia Deckard

We also thank

Rosa & Co at `http://www.rosaandco.com`

and

Bosley LLC at `http://www.clerbos.com/`

who provide considerable feedback on the utility of the software.

Finally, we acknowledge the generous support of the NIGMS grant number: **GM081070** and the PRiME project under subcontract from Mount Sinai which funded the Web application development.