Integrating System Dynamics Models with Online Python-based Analytics

William Glass-Husain
Forio Online Simulations
2601 Mission Street, San Francisco, CA 94110
wglass@forio.com

Abstract

This paper investigates techniques to create online simulations that integrate system dynamics models with other data analytical tools. Specifically it presents a framework for integrating Python and Vensim and briefly discusses sample applications that provide a greater degree of utility for end users than simply allowing an online user to control a pure system dynamics model.

Introduction

The field of System Dynamics has a well-established set of software tools to help individuals model and simulate business and policy problems. Specifically, commercial modeling tools such as Vensim, Powersim, STELLA and others allow users to create and simulate models on their desktop or laptop computer.

In the last 15 years there has been a growing interest in creating simulations based on system dynamics that are distributed via the World Wide Web. (Forio 2016, isee 2016a). Such simulations may be aimed at business school and other higher education programs (Bean 2012), or may be policy oriented simulations in fields such as health care policy or climate change. Such web applications have a broader reach than a purely desktop-based approach as they allow a wide variety of remote users to interact and learn from the model. (Hirsch 2012).

In a separate thread outside of the field of system dynamics, there have been significant advances in the tools available for data analytics. These include both commercial software packages as well as open source languages such as Python or R. (Houghton 2014)

A limitation of many online system dynamic technologies is that they are aimed at system dynamic models only. While there has been work in the community on combining techniques such as optimization, Monte Carlo simulation and sensitivity analysis with system dynamics models, such efforts are usually part of desktop modeling tools and not easy to integrate into online applications.

Using Python with System Dynamics Models

Python has emerged as a frequently cited language for data analytics. It is an expressive language with a variety of available libraries and is relatively easy to learn. Example open source libraries that are freely available to Python users include SciPy (scientific computing), pandas (data and array analysis), and scikit-learn (machine learning). In addition it is straightforward to create Python bindings to other tools which provide an interface in C, making it a lingua franca in the world of data analytics, with most commercial packages providing Python bindings for their applications or libraries as well.

System Dynamics models can be combined with Python-based analysis using multiple methods. As an example, PySd allows SD models to be written directly in Python. (Houghton 2014) Another approach is to take a model developed on a desktop software program and simulate it from Python.

In the remainder of this paper we discuss using Python and the popular system dynamics software package Vensim in an online simulation. Vensim does not directly support Python, but a library is available from Ventana Systems which provides a C interface and hence can be used in Python with a simple interface. The library itself is very C specific, but can be made easier to use with a simple "wrapper" library that is easier to use for those familiar with Python conventions.

Using Python and Vensim Online

A challenge with supporting external users on a central server with a programming language such as Python is preventing the actions of one remote user from affecting another. For example, the individual state of one analysis needs to be completely distinct from the individual state of another analysis. We accomplish this by running a Linux server, with individual user sessions encapsulated in a Docker container. Docker is an open source technology which provides a "sandbox" for individual processes, preventing the user session running within one Docker container from affecting the memory or processing of another.

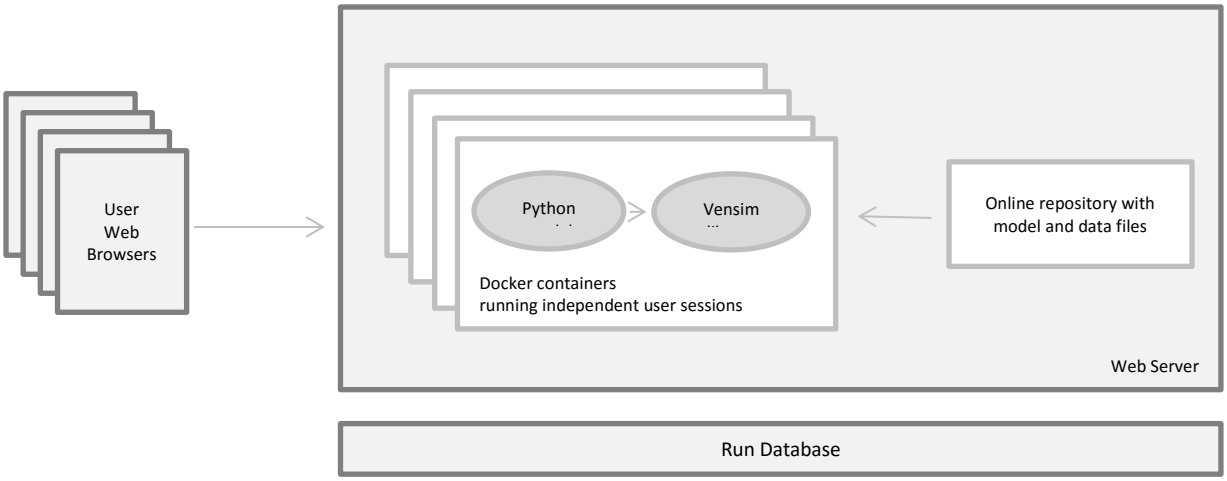

Figure 1: Diagram of web server running multiple user sessions

Example: Sensitivity Analysis

There are a variety of applications that can be developed using a combination of Python and a System Dynamics model. One useful tool is sensitivity analysis.

Consider the classic "Beer Game" model presented below. A common insight gained by studying this model is that changes in the delay time have significant impact on the inventory cost associated with the simulation.

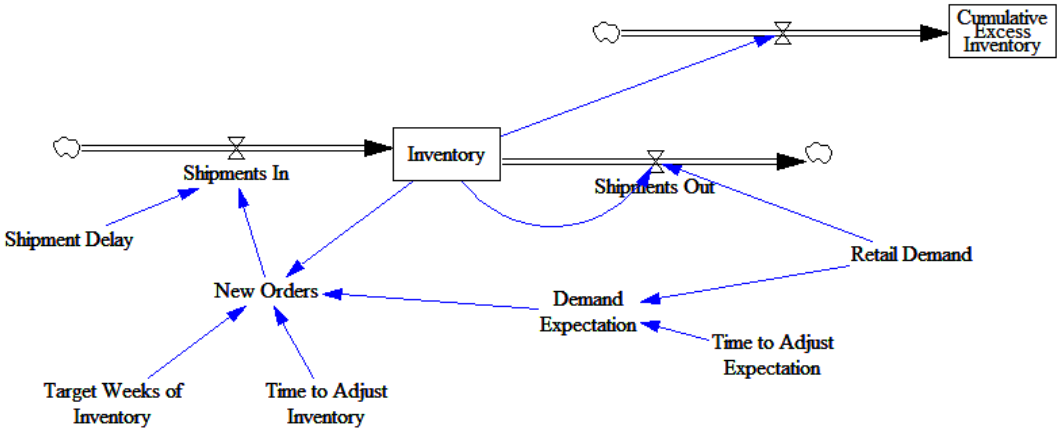

Figure 2: Beer Game model

Many online implementations of this simulation exist, both as a playable game and "what-if" style simulation. Most of these versions play through the simulation from start to finish, presenting results to the user. Users can go go through the simulation after setting specific parameters for Order Delay and Time To Adjust Inventory. Through repeated experimentation the user will be able to see the variation in the results for different patterns of the parameters.

Desktop tools such as Vensim and STELLA offer a sensitivity analysis feature which will automatically do multiple runs, presenting the consequences of different patterns of parameters as a table of values. However, this requires that the user have the actual model in hand as well as access to the modeling software program.

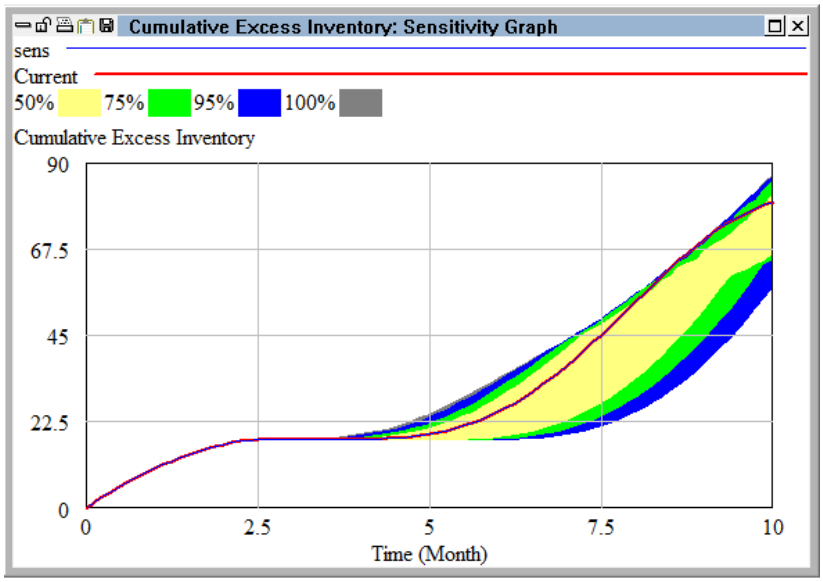

Figure 3: Vensim sensitivity analysis

By using Python, the author of an online simulation can provide this capability to the end user. The user may specify a base set of parameter inputs and choose which ones to vary. After clicking "run" on the web page the server will simulate multiple runs, and then share the results back to the user in a table. For complicated models that take a significant amount of time to simulate, a small subset of the results can be immediately presented on the web page and a more detailed analysis automatically emailed.

```
import numpy as np
from scipy import stats
from vensim_wrapper import VensimWrapper


def sensitivity(model, n, inputs, output_var_name):
    data = np.zeros((n, len(inputs)))
    for i in range(n):
        model.reset()
        for j, (var_name, low_val, hi_val) in enumerate(inputs):
            value = stats.uniform(low_val, hi_val).rvs()
            data[i, j] = value
            VensimWrapper.set_val(var_name, value)
        model.run()
        data[i, -1] = model.get_val(output_var_name)
    return data
```

Figure 4: Python code for sensitivity analysis

Users can gain a deeper understanding of the key leverage points of the simulation from this marriage of a system dynamics model and the sensitivity analysis technique. By including both of these features in a web application the simulation author can help a larger audience gain this intuition instead of the smaller subset with the ability to run the Vensim application directly on a desktop computer.

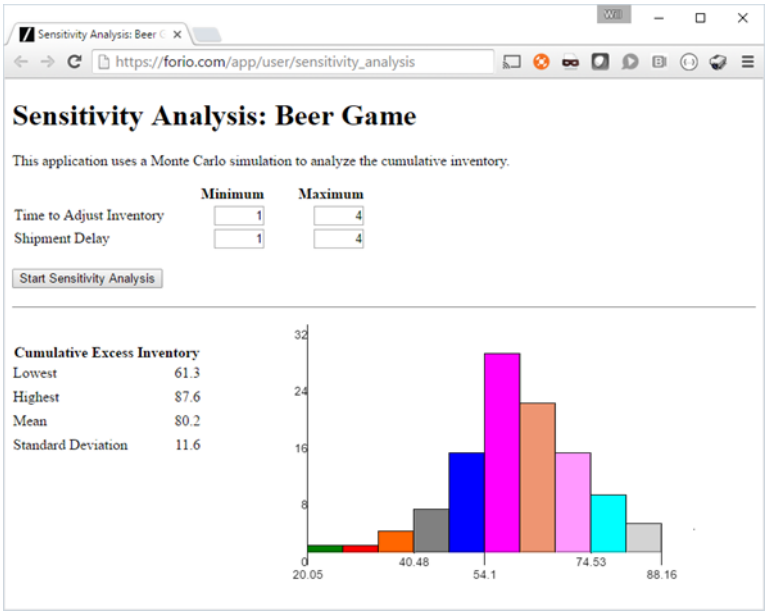

Figure 5: Web application allowing user to run sensitivity analysis

Other Applications

There are a number of other capabilities that can be provided by combining Python and a system dynamics model in an online system. Some examples:

Pre-processing of data. Business and other data is often available at a detailed or transactional level, while system dynamics models usually use data with a longer time scale. A Python wrapper can retrieve data from other services and aggregate it to the appropriate level of detail to fill in the model.

Aggregation of simulation results. It may be useful to aggregate a large set of simulation runs and present the data in the web interface to the user. An educational simulation on the Beer Game may present a summary of results for students playing with different delay times in the supply chain. Monte carlo analysis or sensitivity testing may have aggregate results. Or a correlation report can be done to provide insight into statistical correlation between variables influencing results.

Optimization/calibration. There are a rich variety of optimization libraries available for Python users including commercial tools such as CPLEX as well as numerous open source libraries such as SciPy. Including an optimization capability in an online simulation aimed at an end user allow the user to parameterize a generic model to fit their particular situation.

Combining with other simulation techniques. Models can be combined with other types of simulation including agent-based modeling discrete event simulation. Data may be passed back and forth step by step during the simulation or one simulation may be run to completion and data passed into a different kind of simulation. Again, Python provides a glue that can be used to link different technologies.

Concluding Note

Web simulations that combine system dynamics models with a scientific computing language like Python are valuable. The system dynamics approach provides insight and intuition. Python can provide analytical rigor by running a model repeatedly and analyzing the results with complementary data analytical techniques. The low barriers to using a web-based simulation can permit a broad audience to use and learn from the simulation model and data analysis.

References

Bean M. Developing Business Simulation Games for a Mainstream Audience. 2012. Available at: http://www.systemdynamics.org/conferences/2012/proceed/papers/P1532.pdf

Docker. 2016. https://www.docker.com/.

Forio. Forio Epicenter. 2016. http://forio.com/products/epicenter/.

Hirsch G, Homer J, Milstein B, Scherrer L, Christina I, Landy L, Sterman J, Fisher E. ReThink Health Dynamics: Understanding and Influencing Local Health System Change. 2012. Available at: http://www.systemdynamics.org/conferences/2012/proceed/papers/P1430.pdf

Houghton J, Siegel M, Wirsch A, Moulton A, Madnick S, Goldsmith D. A Survey of Methods for Data Inclusion in System Dynamics Models Methods, Tools and Applications. Working Paper CISL# 2014-03, May 2014. Available at: http://web.mit.edu/smadnick/www/wp/2014-03.pdf

Houghton, J, Siegel, M. Advanced data analytics for system dynamics models using PySD. 2015. Available at: http://www.systemdynamics.org/conferences/2015/proceed/papers/P1172.pdf

IBM. CPLEX Optimizer. 2016. http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/

Pandas. 2016. http://pandas.pydata.org/

Python. 2016. https://www.python.org/

Scikit-Learn. 2016. http://scikit-learn.org/.

SciPy. 2016. http://www.scipy.org/.

isee Systems Inc. 2016a. http://www.iseesystems.com/softwares/NetSimServer.aspx

isee Systems Inc. 2016b. Stella/iThink. http://www.iseesystems.com/.

Ventana Systems Inc. 2016. Vensim. http://vensim.com/.