
How To Run the Eugene-Springfield Model

Center for Urban Simulation and Policy Analysis
University of Washington

May 11, 2008

Daniel J. Evans School of Public Affairs
University of Washington
Seattle, Washington 98195
USA
Email: info@urbansim.org
Web Site: www.urbansim.org

Introduction

This tutorial covers the essentials of running an UrbanSim simulation. We do not cover some complications that are not essential to running UrbanSim, like setting up a database server, creating a base year database, and estimating the model parameters. These omitted items are part of the larger process of preparing to use UrbanSim in a new area, and will be covered elsewhere.

The tutorial uses a new graphical interface that we are developing. This GUI is still in the early stages, and is incomplete in various ways. We'll keep updating the tutorial as additional functionality is implemented.

For the simulation, we use the 1980 base year data for the Eugene-Springfield, Oregon area. This is provided in a non-database format that we refer to as a "cache", since it is a copy of the required data from the associated MySQL database. The base year data includes all the model parameters. Since the model runs quickly on this data, one should be able to compute and visualize indicators from a simulation over 2 years within 20 minutes.

We have tested this tutorial on Windows, Mac and Linux.

Install Software and Download Sample Data

If you have not already done so, install Opus and UrbanSim, along with any needed supporting software, and download the sample data and configurations. Directions for the current version of the code are at <http://www.urbansim.org/docs/installation/>. This tutorial is for the development version of the code currently in the svn repository. It relies on the PyQt package to produce its graphical user interface. Earlier versions (4.1.2 and before) use an older user interface based on the Enthought Traits packages. If you are using a previous release rather than the current version fresh out of the subversion repository, the table of releases at <http://www.urbansim.org/download/> includes a link to the tutorials and installation instructions corresponding to that particular version.

Running a Simulation

In this part of the tutorial, you will execute a two-year simulation run of the Eugene-Springfield metro area.

1. First, run the file `opus.py` in the `opus_gui` directory in your opus workspace. You can do this by navigating to the file, right-clicking on it (on a Mac, control-click), and selecting “Open with ...” and then “python” (Or “python launcher” if you are using a Mac). Alternatively, you can run the file through a graphical editor like Eclipse, or from the command line by typing `python opus.py` from the appropriate directory.
This will launch a graphical user interface that will allow you to, among other things, run a test simulation. If you don’t see the graphical user interface, check your task bar, as the application may be hidden behind another window.
2. In the top menu bar, select “Project” and then “Open Project”. In the projects directory, select the file “eugene_gridcell.xml” and click “Open.” This loads the run configuration information for the Eugene simulation.
3. Select the “Scenario Manager” tab on the left side of the GUI window. You should be able to see and navigate the information loaded from “eugene_gridcell.xml”. Under the top-level node “Eugene_baseline” you will find another called “years_to_run” and under this two variables that specify the start and end years for the simulation. Edit “endyear” by double-clicking in the “Value” column next to “endyear”. Then change the value to “1982”. Many values in the configuration can be edited in this way.
4. Still in the “Scenario Manager” tab, right click on the top-level node “Eugene_baseline” and select “Run This Model”. This will cause a new tab with the name “eugene_gridcell.xml” to pop up on the right side of the GUI. This gets the model ready to run.
5. In the “eugene_gridcell.xml” tab, click the “Start Model...” button. The model will then begin to run. You can watch its progress in the status bar of this tab. When the simulation is finished, the status bar will reach 100% and underneath it will say something like

```
Model finished with status = True
```

That’s it! You have now installed Opus and UrbanSim, a full application for Eugene-Springfield, Oregon, and run a simulation using a default scenario.

The results are all stored on your computer in the `runs` subdirectory of the `eugene` directory where you originally unzipped the cache data. The folder will have a name that indicates when the output was created. If you run this multiple times, it will create a new directory for each run. The contents of these directories are the complete set of “primary” variables and values predicted for all objects in the model, from which UrbanSim can recompute any “computed” variables defined by an Opus . However, these results are in binary files (arrays) that are not easy to read directly. Now we move on to how to examine the results by creating indicators from them. You should keep the GUI open for this next part.

Computing an Indicator

We have now run the simulation, so let’s explore the results. UrbanSim supports the concept of “indicators” that are useful representations of dataset attributes. Conceptually, an indicator is a variable that conveys information on the condition or trend of an attribute of the system considered. When we calculate the indicator with a particular set of data, we produce an indicator result. We can then view the indicator result with an indicator visualization, such as a map, comma-separated-value file, or a tab-delimited file for use by another program.

First, let’s produce a simple map (don’t get too excited, it is only an image map — we’re working on fancier maps with ArcGIS and other systems too, but this is a quick visualizer that doesn’t have all the overhead of a GIS).

1. On the left side of the GUI window, select the “Results Manager” tab. This tab displays a number of indicators and the results of computations done on them. The top-level “Libraries” node has a number of “indicator_library” type subnodes. These nodes represent libraries of indicators. Their subnodes, are the indicators. The top-level “Data_sources” node contains data about the various runs available. It is outside the

scope of this tutorial. The top-level node “Results” has a number of subnodes which are “indicator results” — the results of an indicator computation on particular data.

2. Right-click on the “population” indicator under “Libraries > model.gridcell” and select “Generate results with...” This will open a tab on the right side of the GUI which allows you to specify parameters for generating a particular result.
3. Now we will actually perform the indicator computation. In the “Generate results” tab, choose the value “population” from the “Indicator” drop-down menu. From the “Dataset” drop-down menu, choose “gridcell”, and from the “Source data” drop-down menu, choose “eugene_baseline”. Finally, click the “Generate results...” button. In the log window below the button you will see a small amount of log output about the indicator calculation.
4. Now we display the results. In the “Results Manager” tab, scroll down to the “Results” node. Find the subnode called “population.gridcell.eugene_baseline” and right-click on it. Select “View results as...> Map (Matplotlib)”. This will cause a tab called “gridcell_map_1980_gridcell_population” to pop up in the right side of the GUI. This tab will display the resultant map. *Caution: in this prototype version of the GUI, there are other possible results displayed — these aren’t hooked up to anything yet. Just use the “population.gridcell.eugene_baseline” node.*
5. You can also compute the results at different geographies. In the “Generate results” tab, as before right-click on the “population” indicator under “Libraries > model.gridcell” and select “Generate results with...” Again this will open a tab on the right side of the GUI which allows you to specify parameters for generating a particular result. This time, choose the value “population” as before, but from the “Dataset” drop-down menu, choose “zone” instead of “gridcell.” From the “Source data” drop-down menu, choose “eugene_baseline” as before, and click the “Generate results...” button.
6. Now let’s display the results as a table of zones. (You probably don’t want to have a table of gridcells, since there are so many.) Right-click on the “population.zone.eugene_baseline” node under “Results”. Now select “Table (one per selected indicator)”. A table will pop up on the right side of the GUI. *Caution: right now the map visualization won’t work for zones, just gridcells — so for zones just view these as a table.*

Closing Comments

At this point, you have run a simulation and generated and visualized a couple of indicators from the simulated results. You may have also exported the UrbanSim database for a year and examined it in another system. This tutorial has just scratched the surface of what you can do with UrbanSim, of course. For more information, see the accompanying Reference Manual and User Guide. In addition, we plan to expand the content in this tutorial and add additional tutorials as we get time and as the functionality develops. We would welcome user contributed tutorials also!

If you have any particular requests for documentation, let us know by emailing the users@urbansim.org email distribution list.

Acknowledgements

This tutorial was put together by the UrbanSim project team in response to various requests for a sample application that would allow quickly installing UrbanSim, running an application, and examining the data, and has been tested by several users.

A special thanks is in order to Bud Reiff, Bob Denouden and others at the Lane Council of Governments for their generosity in sharing the data we use in this tutorial. This research has been funded in part by grants from the National Science Foundation (IIS-0534094 and IIS-0705898), and in part by a grant from the Environmental Protection Agency (R831837). We’d like to thank several users as well for their financial support of the development of UrbanSim and OPUS.