UrbanSim2: Simulating the Connected Metropolis

Fletcher Foti
University of California, Berkeley (USA)

Paul Waddell

UrbanSim2: Simulating the Connected Metropolis

by Fletcher Foti and Paul Waddell

University of California, Berkeley (USA). fscottfoti@gmail.com

Abstract

UrbanSim is an open source software platform for agent-based geospatial simulation, focusing on the spatial dynamics of urban development. Since its creation UrbanSim has been used in the official planning processes for at least a dozen regional governments which were used to help allocate billions of dollars in regional investments in transportation infrastructure.

UrbanSim was first conceptualized in the late 1990's and implemented using the Java programming language. The technology landscape for scientific computing changed dramatically after that, and by 2005 UrbanSim was converted to Python, making heavy use of Numpy to vectorize calculations. By 2014, it became clear that UrbanSim should be reimplemented again to take advantage of significant advances in the libraries available for scientific Python. The new version of UrbanSim, called UrbanSim2, makes extensive use of community-supported scientific Python libraries to reduce the amount of domain-specific customized code to a minimum.

UrbanSim is an excellent case study for the power of leveraging the work of the scientific programming community as scaffolding for a domain-specific application, as opposed to building an extensive customized solution in each domain. Additionally, the open and participatory nature inherent in nearly all of the open source projects described here has been particularly embraced by governments, who are often reticent to support large commercial institutions and balkanized and private data formats and software tools.

Keywords: Open Source; Regional Planning; City Planning; Transportation Planning;.

1 Introduction

UrbanSim is an open source software platform for agent-based geospatial simulation, focusing on the spatial dynamics of urban development. It simulates the choices of locations of households and businesses in a metropolitan region in order to predict demand for public infrastructure such as transportation, energy and water. It has been most widely used for regional transportation planning, to assess the impacts of transit and roadway projects on patterns of urban development, and the indirect effects these have on travel demand. In recent years, urban models are increasingly used to understand how to reach sustainability goals, including reducing resource use and land consumption, and how best to substitute sustainable modes like transit, walking, and biking for increasing automobile use.

UrbanSim was first conceptualized and implemented almost 15 years ago (Waddell, 2000, 2002). The initial implementation was in Java, and for performance reasons it used an approach of 'exploded objects' to represent the millions of agents in its simulation, to minimize object overhead (Noth et al., 2003). By 2005, a decision was made to re-implement the UrbanSim platform in Python, taking advantage of the rapid advances made in the scientific Python community, most notably Numpy for multi-dimensional array computations. This version was referred to as the Open Platform for Urban Simulation (OPUS), and intended to stimulate broad collaboration as an open source project among international research teams working on urban modeling (Waddell et al., 2005).

Since its creation and release on the web as an open source project in 1998, UrbanSim has been increasingly used in the official planning processes for at least a dozen Metropolitan Planning Organizations (MPOs), which use UrbanSim and their travel model platforms to evaluate the planning of billions of dollars in regional investments in transportation infrastructure. By the end of its first ten years, UrbanSim had become the most widely used land use modeling platform for regional planning by MPOs (Lee, 2009), although a variety of other models have been used by MPOs, including 'home-grown' models (Hunt and Abraham, 2005; Wegener, 2004).

UrbanSim has grown over the past two decades to become a robust open source software platform, and its history closely parallels the history of many other open source projects, including the Python libraries on which it has grown to depend. Python was introduced by Guido van Rossum in 1991 and began to rapidly gain popularity. A key Python li-

brary -the vector and matrix manipulation library NumPy -was first released in 2006. When UrbanSim was reimplemented in Python beginning in 2005, a great deal of code was created using Numpy to build statistical models for estimating the parameters of regression and discrete choice models, and for efficiently simulating using the fitted models. In order to manage data effectively, the research team developed a heavyweight DataSet Class, to make sets of Numpy arrays behave more like relational database tables, with relational joins and a variety of query operations. In order to make the code more accessible to modelers, a domain-specific expression language was created (Borning et al., 2008). And in order to create a graphical user interface, an extensive PyQt infrastructure was built to auto-generate GUI elements from an XML file manipulated by the GUI. In short, a large volume of domain-specific, customized code accumulated as the project met a variety of needs. It eventually grew to over 150K lines of code, and debugging became more complex due to the many levels of indirection in the Python tracebacks whenever an error occurred. In other words, UrbanSim had become a large domain-specific application with a large customized code base to solve several of the problems that were eventually taken up by the scientific programming community as a whole. As a result, it was becoming a challenge to maintain.

In 2010, the Pandas data analysis library was released. Its user community has grown rapidly, and it has become a standard part of many scientific Python bundles. Pandas has implemented in a more general way most of the functionality that the UrbanSim team had developed in the UrbanSim DataSet class and the OPUS expression language. The new version of UrbanSim, called UrbanSim2, is the result of a reassessment of the landscape of available scientific Python libraries, and completely eliminates most of its legacy customized data management and expression language code, and its customized graphical interface. It replaced the UrbanSim DataSet class and expression language with Pandas, and replaced its GUI with Python, at least initially, with ongoing experimentation in web-based interfaces. The implementation of UrbanSim has thus again gone through a massive transformation in its software implementation, making an excellent case study for the power of leveraging the work of the scientific programming community to provide the scaffolding for a domain-specific application, rather than building and maintaining a heavy-weight customized solution in each domain.

This paper will outline this process and the argument for refactoring large domain-specific applications to rely more extensively on well-supported open source libraries such as exist in the scientific Python community. We begin by giving a brief history of urban modeling, followed by a discussion of the theory and implementation needs of UrbanSim, a description of the implementation efficiencies gained between the two versions of UrbanSim made by leveraging current open source geospatial projects, and closing with a discussion of future work on the topic.

2 A Brief History of Urban Modeling

Urban modeling began as economic theory, as early as 1826 in Von Thünen's "The Isolated State" (translated in Von Thünen and Hall, 1966), in which he outlined how in a "featureless plain" different crops would be grown in concentric rings around the city depending on their market value and cost of transportation. High value crops like vegetables and also heavy crops like firewood would both be grown near to the town while grain and ranch animals could be grown far from the city.

Walter Christaller then expanded these ideas to the urban context in Central Place Theory (Christaller, 1968) by proposing that while some products known as comparison goods —e.g. automobiles or appliances— would be consumed only rarely, other products known as convenience goods —e.g. food— would be consumed repeatedly. He proposed a geometry within the city of nested and overlapping hexagons in which vertices are shopping nodes and convenience goods would occur roughly 6 times as frequently within the city as comparison goods.

The modern era of urban models probably began with the exploration of the idea that increased transportation access leads to increased development intensity (a centralizing force), and that people tradeoff increased transportation costs to consume more housing (a de-centralizing force) (Hansen, 1959; Alonso, 1964). This presaged the field of transportation-land use interaction in which researchers explore how the built environment affects how people travel. Seminal work in the field shows that density, diversity, and design of the built environment of our cities all impact how far people will travel and which modes they will take, with more dense and diverse environments encouraging sus-

tainable modes like walking and transit (Kockelman and Cervero, 1997; Ewing and Cervero, 2001).

Almost as soon as people understood that the built environment has a strong effect on how people travel, academics and transportation planners began to model the transportation systems of entire urban regions in order to predict travel patterns under counterfactual situations, like building a new highway or transit line, or accommodating an increase in population. The most prevalent methodology still in use today is the 4-step model (de Dios Ortúzar et al., 2001), in which the four steps are: trip generation, trip distribution, mode choice, and route choice. This framework runs four statistical models in sequence, answering for each person how many trips to take, where to take them, on which mode (auto, transit, walk, etc) and by which route.

Eventually planners began to ask more complicated questions of the models, including the impact of toll lanes and bridges, the effect of changing transit service characteristics, the result of carpooling and household coordination, and many other highly detailed policy questions. This gave birth to the modern advanced transportation models which fall under the rubric of activity-based travel models (ABMs) first proposed by Ben-Akiva and Bowman (Ben-Akiva et al., 1998; Bowman and Ben-Akiva, 2001).

In this paradigm, every person in a region is modeled as they move through the simulated day, sometimes for time increments as small as 5 minutes, capturing where each person is, what they are doing, and how they move from place to place. The dominant methodology in this framework is to run discrete choice models in sequence; the Portland implementation of the Ben-Akiva and Bowman framework has five levels of hierarchical choices: activity-patterns, time-of-day, mode-destination, sub-tours and intermediate stops.

Although the methodology employed by UrbanSim will be discussed in detail in the next section, the theory proposed therein owes direct lineage to this history of transportation models in the literature. UrbanSim also simulates a number of statistical models in sequence, using many of the same methods as the ABMs. In point of fact, many of the most advanced cities run both ABMs and land use models, using land use models to predict the spatial distribution of households and jobs (and other economic, environmental and social indicators) and the ABMs to predict the demand for transportation infrastructure and other travel characteristics.

Most urban modeling is performed at the level of regional government (i.e. MPOs), although large cities sometimes also have implementations of the models described here. MPOs are regional governmental bodies that were formed by the 1962 Federal-Aid Highway Act whose main purpose is to create and implement a long-term (typically 30 year) vision for the transportation infrastructure of a region, balancing the needs of constituent cities, as well as environmental and social equity considerations, while meeting demand for new highways and transit lines and maintaing existing infrastructure.

In California, state law SB-375 provides ground breaking legislation that due to the interconnected nature of land use and transportation (Barbour and Deakin, 2012), requires all MPOs to coordinate planning of land use and transportation infrastructure and to implement both land use models and transportation models. California MPOs are thus home to some of the most advanced urban models that exist today.

3 The Theory of UrbanSim

UrbanSim is built from several individual models of urban behavior. The models are typically statistically estimated, but this is not an essential requirement. In fact, UrbanSim can be viewed as a batch data analysis process with separate modules, each representing a specific urban behavior, and each module is allowed to read and write to the set of available urban data which includes at minimum: parcels for spatially subdividing land, the buildings which exists on those parcels, and the households and jobs which occupy that built space. The simply defined purpose of UrbanSim is to predict the spatial distribution of households and jobs in a future year, with an accurate representation of where new buildings will be built.

UrbanSim at its core is four models -the price model, location choice model, transition model, and real estate development model. Residential price models are called hedonics (Rosen, 1974; Waddell et al., 1993) and are usually linear regressions where the dependent variable is price (or a transformation of price) and independent variables typically include square footage, lot size, number of bedrooms and bathrooms, and attributes of the neighborhood like average income, regional accessibility by automobile and transit, local accessibility by walking, and others. A residential location choice model (McFadden, 1978; Lee et al., 2010) is a logit models where the number of alternatives is discrete and often quite

large. For instance, households might choose from among all of the neighborhoods in a region in their choice of residence. Transition models are used to change the demographics of the population, including aging, family formation and separation, as well as births, deaths, and migration from other regions. As most regions in the United States are growing, the challenge becomes housing all the new households, and thus the creation of new residential buildings must also be modeled accurately.

The real estate development model is an extremely specialized model in Urban-Sim and is used to capture real estate developer behavior and analyze the cash flow of potential developments for profitability. This is called a pro forma (Miles et al., 2000; Brueggeman and Fisher, 1997) and is traditionally performed in a spreadsheet program, but UrbanSim uses Python to perform millions of pro formas for a large region to analyze the profitability of a multitude of possible buildings. Inputs to pro formas are rents or prices by unit type (1 bedroom, 2 bedroom, etc), construction costs per square foot, prevailing interest rates (and forecasts for future interest rates), the rate at which households will occupy a new development (called absorption), and jurisdictional policies including zoning restrictions, affordable housing policies, and parking requirements.

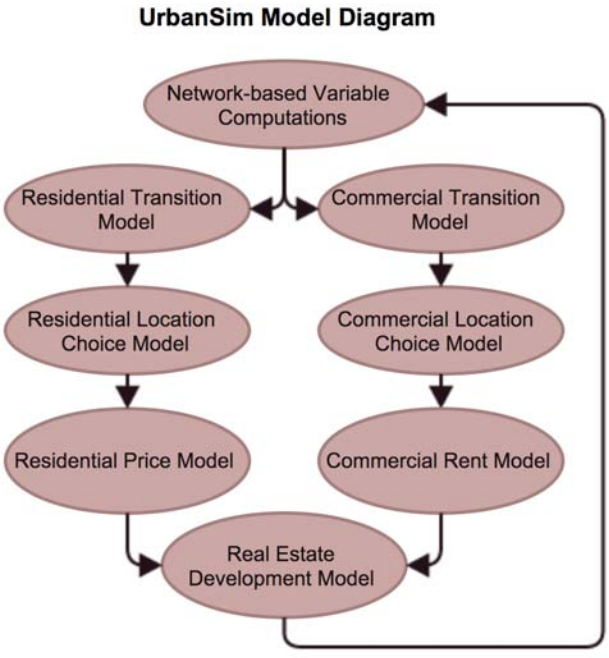

Figure 1: Diagram of the simplified UrbanSim model system.

Although the models above are described in detail for the residential/housing model set, there are analogous models for commercial entities including rent models for retail, office, and industrial building types, location choices for jobs by employment sector, predictions of job growth and decline, and the creation of new commercial buildings. The first three models are generally independent for the residential and commercial model sets (although aggregations of access to commercial uses might be variables in the residential models and vice versa). The real estate development model requires coordination between the residential and commercial markets; any parcel which is zoned to allow several potential uses must make a choice among alternatives based on the relative profitability of producing a building of a given type. Thus different uses compete for land with the highest profitability uses outcompeting less profitable uses. A diagram of the UrbanSim system of models is shown in Figure 1.

4 Network-based Spatial Variables

The selection of geography is enormously important in understanding any urban behavior. For instance, variables that are predictive of home prices and residential location choices can include the boundaries of school districts, other public goods provided within the boundary of a city (e.g. police protection), and these are large geometric shapes that are well-defined in the region. On the other hand, variables used to describe a person's perception of his neighborhood are not as easily defined, and much research has been performed to understand how people interpret their surrounding areas (Guo and Bhat, 2007; Grannis, 1998).

Although it is standard practice to use large polygons like census tracts, city boundaries, zip codes, etc to provide mutually exclusive boundaries for aggregations in the city, nonetheless this approach has clear weaknesses as polygon definitions are subject to judgement, they exhibit boundary effects (e.g. an element is either fully included or not included in an aggregation), and almost no spatial process will be completely homogeneous within such a polygon boundary. This can lead to the well documented MAUP (Modifiable Areal Unit Problem) (Openshaw, 1984), which is a potential issue for almost all of the spatial models used by UrbanSim.

To avoid this problem, UrbanSim now uses a framework for quantifying urban space where the city is represented as land use spatially located

within a multi-modal transportation network (Foti, 2014). In this formulation, all land use is allocated to the nearest street intersection of the local street network and then aggregations can be performed within buffers surrounding every origin street intersection in the network. Figure 2 shows how parcels of land are each mapped to their nearest street node. In this figure, points represent intersections, lines represent the local streets obtained through OpenStreetMap, and parcels are assigned the color of the nearest street intersection.

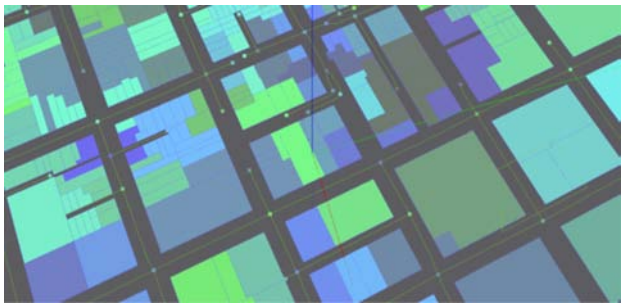

Figure 2: Map of parcels to street nodes (mapping indicated by color) .

Aggregations are performed to a user-specified horizon distance along the network with a user-specified decay so that items that are further away are weighted so that they affect the origin location less, consistent with Tobler's second law of geography (Tobler, 1970). Aggregations can be any of the standard statistical measures including max, min, mean, standard deviation, and sum. Networks are also fully abstracted so that where information is available, local street networks, transit schedules, or congested (i.e. accounting for traffic) automobile travel times can be used to perform aggregations (see Foti, 2014 for more details on how these computations are performed). Thus variables can be computed such as "average income within 500 meters" or "jobs in the technology sector within 45 minutes transit ride," and many others.

The current implementation leverages the high performance network algorithm Contraction Hierarchies (Geisberger et al., 2008) and is written in C and fully multithreaded so that aggregations are computed very quickly. Local-scale aggregations (akin to WalkScore) can be performed for every street intersection in the United States in about 12 seconds. For a large region like the Bay Area, aggregations can be performed for all of the 226 thousand street nodes in a small fraction of a second.

The advantage of this "street node geography" is that its geometric definition is an emergent property of the local street network, which has a very real physical manifestation (i.e. is not subjectively defined). Additionally, the distance between intersections is usually small in dense urban areas which need to be represented most accurately, thus the land mapped to a street node is far more likely to be homogenous than the land within a geography as large as a census tract. Finally, network aggregations are overlapping and a decay is applied so that there are no boundary effects. Although representing actual access and egress points for parcels on the street network is possible, this information is not easily obtainable at this time, and street node geography provides a reasonable compromise in accuracy and also provides an order of magnitude increase in performance.

The current implementation does have a few limitations, including the lack of information about local-scale pedestrian access, sidewalks, and street crossings, information on the qualitative aspects of the pedestrian environment, etc, but is nontheless an extremely efficient substitute for polygon-based aggregations in GIS. The use of these variables is now ubiquitous in new UrbanSim implementations, and typically all network aggregation variables needed to run models are computed at the beginning of each simulated year so that all subsequent models will have access to these variables.

Although these metrics are extremely useful as independent variables in urban statistical modeling, it is hoped that this framework will be generally useful as a method for visualizing spatial data with far more precision than is typical with large polygonal geographies. For example, Figures 3 and 4 show the average home sales price in the Bay Area using zonal aggregations (Figure 3) and aggregations along the local street network (Figure 4). Although the color scale is discrete, the actual values for the aggregation are continuous and smooth for the network aggregations, and discontinuities are easily visible with the polygonal boundaries.

5 Leveraging Open Source Tools

UrbanSim was first translated to the Python programming language in 2006. In the intervening decade, the available technology and best practices for a large software project have changed dramatically. Innovations of particular relevance to the new UrbanSim implementation include:

- Python has added numerous supporting li-

Figure 3: Residential rent in the Bay Area aggregated by zone.

Figure 4: Residential rent in the Bay Area aggregated by network buffer queries.

braries including the Pandas data analysis package, Patsy, and SciKits and StatsModels for statistical analysis

- XML has been replaced by JSON for specifying key-value configuration documents
- The web has become the ubiquitous technology for graphical interfaces with advances including Angular, Leaflet, D3, Bootstrap, and others
- Github has become a free (for public projects) online collaborative tool ideal for large distributed code projects of this sort
- Anaconda now provides a well-tested Python distributions with numerous included Python packages to reduce installation headaches
- Spatial databases like PostgreSQL/PostGIS and the OSGeo packages have made GIS functionality available without dependency on commercial software packages like ArcGIS by ESRI

As the technology landscape has changed so dramatically, it became clear that UrbanSim would need to be overhauled substantially. In particular, the original UrbanSim was written in the year 2006, and the same needs that drove the eventual creation of the Pandas data analysis library were also present in the UrbanSim project. An abstraction layer was required to overlay a NumPy columnar datastore with names and types, the ability to read and write multiple formats of data, to merge/join different datasets, and to perform various aggregations across categorical variables to compute sums, means, and other typical statistical metrics. These operations became the basis for the UrbanSim expression language (Borning et al., 2008) and the similarities to the eventual Pandas package (McKinney, 2012) are many. Pandas has now been widely embraced in the Python community, with over 200 contributors, 9,000 commits, and at the time of this publishing 88,000 downloads per month.

The Pandas project is but one example of the integration with the larger open source software community that needed to take place, and so it was decided that UrbanSim would be re-implemented from scratch to work directly with Pandas, StatsModels, SciKits, etc. Although many of the more nuanced behavioral models have not yet been ported to the new framework, the bulk of the work has now been completed, and the new models have been used in active planning processes in the regions of Denver and Paris, and implementations are currently underway in many other regions. It should be noted that the use of well-tested community-supported frameworks has reduced the code complexity from over 100,000 lines of code to only 4,039 lines at the time of this writing. The new version of UrbanSim is now supported by the company Synthicity and is available as open source software under the AGPL license at https://github.com/synthicity/urbansim.

6 Statistical Model Configuration using JSON

Viewed as described in the previous sections, UrbanSim is essentially a configuration system for Pandas variables and statistical models provided by StatsModels. In fact, a small handful of parameters can describe each model configuration. In this section, the residential price hedonic will be used as the canonical example, and below is the list of parameters necessary to specify such a model:

- A small code wrapper is required to describe where the data comes from. Currently, tabular data is stored in the HDFStore as is common with Pandas.
- The main table for estimation/simulation must be specified. The canonical example would be a data table of home prices with sales prices and attributes of the home including square footage, lot size, number of bedrooms and bathrooms, etc.
- Additional tables may be merged/joined to the main table. Frequently a merge must be performed between the estimation table and the dataset that results from the set of network-based aggregations that are described in the previous section.
- If necessary a few lines of Python/Pandas can be used to transform variables.
- Filters can be applied to remove obviously incorrect or degenerate rows of data.
- The model must actually be specified. This is usually done with Patsy, which is a highly parsimonious R-style syntax for specifying the dependent and independent variables from the dataframe generated by the steps above.

- The results must be saved to an output data store, for both estimation and simulation. For estimation, coefficients on variables must be saved and for simulation the predicted output variables are saved for use in subsequent models.

UrbanSim has been designed to take key-value pairs which specify the above set of parameters in a JSON format. An example configuration is shown in Figure 5 which gives the configuration for a linear regression on home prices and shows the parsimony of specifying a statistical estimation in this way. Each model can be specified with 10-15 key-value parameters and then models can be executed in sequence to create a simulation of the full regional real estate market (as shown in Figure 1). It is possible that this sort of framework can be expanded to be useful to the broader community of StatsModels users, but this task remains for future work.

```
{
    "model": "hedonicmodel",
    "output_table": "dset.buildings",
    "add_constant": true,
    "internalname": "buildings",
    "patsy": [
        "I(year_built < 1940)",
        "I(year_built > 2005)",
        "np.log1p(stories)",
        "ave_income",
        "poor",
        "jobs"
    ],
    "merge": {
        "table": "dset.nodes",
        "right_index": true,
        "left_on": "_node_id"
    },
    "output_varname": "nonresidential_rent",
    "table": "dset.costar",
    "table_sim": "dset.building_filter(residential=0)",
    "dep_var": "averageweightedrent",
    "segment": [
        "general_type"
    ],
    "dep_var_transform": "np.log",
    "output_transform": "np.exp"
}
```

Figure 5: A sample JSON configuration used to specify a residential sales price model.

Once models are configurable in JSON, and given that JSON is the vernacular for client-server communication on the internet, it is an incremental step to create a web service which, after specifying an HDFStore from which to read all necessary data, models can then be estimated or simulated by making http requests with a JSON model specification. A simple website has been created to read, write, and edit JSON specifications, to run sets of models in sequence, and to create charts of model results using D3 and maps of model results using Leaflet. Basic browsing of tables in the HDFStore is also available. Thus a graphical interface is underway which can be used to configure and run statistical models via a website and even to run data analysis batch jobs, and a screenshot is shown in Figure 6.

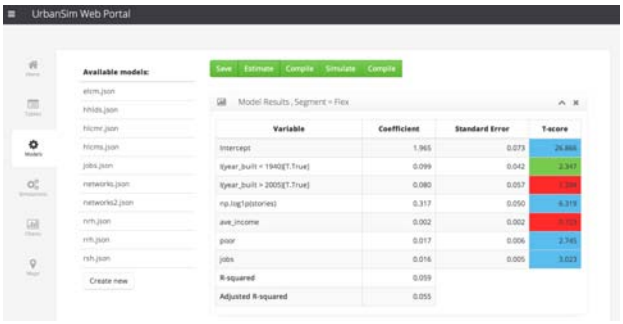

Figure 6: Screenshot of the current UrbanSim web portal.

It is worth noting that the one area that community-based scientific programming tools have been insufficient is for discrete choice multinomial (MNL) models where the number of alternatives is extremely large. For running MNL models where the number of choices is small -e.g. the choice of travel mode between auto, transit, and car -the models provided in StatsModels are sufficient. However, for location choice models in a large region, the number of alternatives can be in the thousands or even hundreds of thousands, and a probability may need to be computed for each of the alternatives.

For this case, special purpose models and model estimation code have been written which take the same basic form as the StatsModels model interface, but are not currently included in the StatsModels distribution. It is unclear at this time if these models are useful to the broader scientific community or if they are only of importance to the urban modeling community.

7 Conclusion

This paper has presented the history of UrbanSim in the context of the history of the open source projects which support it. Numerous advances have been made in Python, including Pandas and StatsModels, ease of coordinating distributed code projects using GitHub, distribution and installation of Python

projects using Anaconda, and new interface technology including JSON, Angular, D3, and Leaflet. All of these advances have enabled UrbanSim to make an evolutionary leap and minimized the amount of code necessary to create a domain-specific application.

Additionally, embracing the larger open source geospatial community has allowed UrbanSim a competitive advantage over other projects which have not embraced the available tools with the same alacrity. In fact, the methodology described in this paper has now convinced the Association of MPOs to begin work on a prototype which expands the use of this methodology to activity-based travel models which are critical to the transportation planning operations in dozens of the larger regions in the United States and internationally.

The open source and community-supported nature of the core projects, including UrbanSim -and in particular the open and collaborative nature of online tools like GitHub -are garnering a positive response from proponents of open and accountable government as well as groups which support the transparency of large agent-based simulations used in governmental processes. The grassroots nature inherent in nearly all of the open source projects described here has been particularly embraced by governments, who are often reticent to support large commercial institutions and balkanized and private data formats and software tools. Clearly open source tools are well suited for applications within government, and UrbanSim is an excellent case study for progress that can be made to this end.

References

Alonso, W. (1964). Location and land use: toward a general theory of land rent. Harvard University Press.

Barbour, E. and Deakin, E. A. (2012). Smart growth planning for climate protection. Journal of the American Planning Association, 78(1):70–86.

Ben-Akiva, M., Bowman, J. L., et al. (1998). The day activity schedule approach to travel demand analysis. PhD thesis, Massachusetts Institute of Technology.

Borning, A., Ševcıková, H., and Waddell, P. (2008). A domain-specific language for urban simulation variables. In Proceedings of the 2008 international conference on Digital government research, pages 207–215. Digital Government Society of North America.

Bowman, J. L. and Ben-Akiva, M. E. (2001). Activity-based disaggregate travel demand model system with activity schedules. Transportation Research Part A: Policy and Practice, 35(1):1–28.

Brueggeman, W. B. and Fisher, J. D. (1997). Real Estate Finance and Investments. Inc.

Christaller, W. (1968). Die zentralen Orte in Suddeutschland.

de Dios Ortzar, J., Willumsen, L. G., et al. (2001). Modelling transport. Wiley.

Ewing, R. and Cervero, R. (2001). Travel and the built environment: A synthesis. Transportation Research Record: Journal of the Transportation Research Board, 1780(-1):87–114.

Foti, F. (2014). Measuring the Contribution of Walkable Amenities to Home Values and Residential Location Choices. DRAFT, University of California, Berkeley.

Geisberger, R., Sanders, P., Schultes, D., and Delling, D. (2008). Contraction hierarchies: Faster and simpler hierarchical routing in road networks. Experimental Algorithms, pages 319–333.

Grannis, R. (1998). The importance of trivial streets: Residential streets and residential segregation 1. American Journal of Sociology, 103(6):1530–1564.

Guo, J. Y. and Bhat, C. R. (2007). Operationalizing the concept of neighborhood: Application to residential location choice analysis. Journal of Transport Geography, 15(1):31–45.

Hansen, W. G. (1959). How accessibility shapes land use. Journal of the American Institute of Planners, 25(2):73–76.

Hunt, J. D. and Abraham, J. E. (2005). Design and implementation of PECAS: a generalised system for allocating economic production, exchange and consumption quantities.

Kockelman, K. and Cervero, R. (1997). Travel demand and the 3Ds: density, diversity, and design. Transportation Research Part D: Transport and Environment, 2(3):199–219.

Lee, B. H., Waddell, P., Wang, L., and Pendyala, R. M. (2010). Reexamining the influence of work and nonwork accessibility on residential location choices with a microanalytic framework. Environment and Planning A, 42(4):913–930.

Lee, D. (2009). 2009 TMA MPO modeling activity survey. Technical report, Fredricksburg Area MPO.

McFadden, D. (1978). Modelling the choice of residential location. Institute of Transportation Studies, University of California.

McKinney, W. (2012). Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython. O'Reilly Media.

Miles, M. E., Berens, G., and Weiss, M. A. (2000). Real estate development: principles and process. Urban Land Institute Washington, DC.

Noth, M., Borning, A., and Waddell, P. (2003). An extensible, modular architecture for simulating urban development, transportation, and environmental impacts. Computers, Environment and Urban Systems, 27(2):181–203.

Openshaw, S. (1984). Ecological fallacies and the analysis of areal census data. Environment and Planning A, 16(1):17–31.

Rosen, S. (1974). Hedonic prices and implicit markets: product differentiation in pure competition. The journal of political economy, 82(1):34–55.

Tobler, W. R. (1970). A computer movie simulating urban growth in the detroit region. Economic geography, 46:234–240.

Von Thünen, J. H. and Hall, P. G. (1966). Isolated state: an English edition of Der isolierte Staat. Pergamon Press.

Waddell, P. (2000). A behavioral simulation model for metropolitan policy analysis and planning: Residential location and housing market components of Urban-Sim. Environment and Planning B: Planning and Design 2000, 27(2):247–263.

Waddell, P. (2002). UrbanSim: modeling urban development for land use, transportation, and environmental planning. Journal of the American Planning Association, 68(3):297–314.

Waddell, P., Berry, B. J. L., and Hoch, I. (1993). Residential property values in a multinodal urban area: New evidence on the implicit price of location. The Journal of Real Estate Finance and Economics, 7(2):117–141.

Waddell, P., Ševcıková, H., Socha, D., Miller, E., and Nagel, K. (2005). Opus: An open platform for urban simulation. In Computers in Urban Planning and Urban Management Conference, London.

Wegener, M. (2004). Overview of land use transport models. Handbook of transport geography and spatial systems, 5:127–146.