Reading Resources and Computing Tools for Research using Agent-Based Modeling

by Jonathan Gilligan

EES 4760/5760: Agent- and Individual-Based Computational Modeling
August 20, 2025

Contents

This handout provides recommendations of reading resources (both in print and on the web) and open-source computing tools relevant to agent-based modeling. This course just scratches the surface of what agent-based and individual-based models can do, and this handout points you toward resources that will be useful if you are interested in pursuing agent-based and individual-based modeling beyond what we cover this term.

Reading Resources

Foundations of Agent-Based Modeling

These are books that cover the foundations of agent-based modeling. Many of them are written by pioneers in the field.

Popularizations

These two books were written by pioneers of agent-based modeling, but are aimed at explaining their discoveries to the general public. Both are well written and enjoyable to read. They focus on describing what the authors did in general terms, rather than in the kind of technical detail that you would find in scholarly writing.

- Thomas C. Schelling, *Micromotives and Macrobehavior* (Norton, 2008). Thomas Schelling, who died in December 2016, won the Nobel Prize in economics for his work on game theory and avoiding nuclear war. He also developed a model of racial segregation in housing that many consider to be the first true agent-based model. He wrote this model in 1971, and used used coins on a checkerboard to represent the agents. A version of his model is included in the

NetLogo model library. The focus of this book is on *emergence*: How large-scale phenomena ("macrobehavior," in Schelling's termiology) can emerge unexpectedly from individuals pursuing activities in which they only consider things in their immediate vicinity ("micromotives") without considering the larger picture.

- Robert Axelrod, *The Evolution of Cooperation* (Basic Books, 2006). Axelrod is a political scientist who studies game theory and conflict. In 1981, as computers were first becoming wide-spread in social science research, he held a tournament in which he asked people to submit computer programs to play a simple two-player game called "The Prisoner's Dilemma." Many of these programs were very complicated, but to Axelrod's surprise, the clear winner was a simple strategy called "tit for tat" that always played the same move the opponent had played in the previous round. This strategy implied that in a society or ecosystem where many individuals used a strategy like tit-for-tat in their interactions with each other, there would be a strong evolutionary advantage to cooperating with others, rather than trying to take advantage of them. This book describes the tournament and its implications both for evolutionary biology and also for understanding morality and the foundations of civilization.

Technical Books

These books are written for a more technical audience, so they discuss agent-based modeling in greater detail and are more challenging to read, but they are also very accessible and trace agent-based modeling from its origins to the present day.

- Joshua M. Epstein and Robert Axtell, *Growing Artificial Societies: Social Science from the Bottom Up* (MIT Press, 1996). In the 1970s and 80s, a few isolated researchers, such as Schelling and Axelrod, had conducted rudimentary research using primitive agent-based models, but Epstein and Axtell revolutionized the field in the mid-1990s by showing that agent-based models could be used to explore a wide range of social-science research questions. This book introduces Epstein and Axtell's "Sugarscape" model and shows how they used it to study all sorts of questions, including the distribution of wealth in a market economy (including a sensitivity analysis for the conditions that are necessary for a market-based economy to produce Pareto-optimal outcomes), evolution of traits in biological systems, spread of diseases in epidemics, and outbreak of conflict and war. This book also lays out a methodology for how to use agent-based models to conduct research, and how to systematically explore the emergence of macrobehaviors from micromotives.

- Joshua M. Epstein, *Generative Social Science: Studies in Agent-Based Computational Modeling* (Princeton University Press, 2006). This book picks up a decade after *Growing Artificial Societies* and discusses challenges of trying to model the real world in detail with agent-based models (as opposed to the schematic and stylized character of Sugarscape). The book covers several case studies, including an agent-based model that Epstien and others used to study the collapse of Anasazi settlements in Long House Valley, Arizona, around 1350 CE. This model draws on detailed archeological and paleoclimatological data and the book compares the model output with actual historical records of population in Long House Valley from 800–1400 CE. The book also describes models of patterns in retirement in the United States in the 1960s–1980s; emergence of racial and cultural conflict and outbreaks of violent conflict; and management of disease epidemics and bioterror attacks.

- Joshua M. Epstein, *Agent Zero* (Princeton University Press, 2014). Previous work on agent-based modeling focused on agents as rationally trying to pursue some objective, such as maximizing their life or health. Here, Epstein introduces a completely different approach, in which agent behavior is rooted in principles from empirical neuroscience to account for the role of emotions, especially fear, in influencing decisions. In this book, he examines a range of phenomena including war-time atrocities, jury deliberations, and economic booms and busts. All of the models in this book are implemented in NetLogo and Epstein provides the source code at the book's web site `http://press.princeton.edu/titles/10169.html`.

On-line resources

Model libraries and useful web sites

- The CoMSES OpenABM library `https://www.comses.net/codebases/` has a collection of well-documented open-source models, many of which were used in peer-reviewed research publications. You can search this library by keyword, so it can be a very good resource when you are thinking about research projects and want to see what other people have done on similar problems. It also hosts an online textbook about agent-based modeling called *Games and Gossip*, by Marco A. Janssen (`https://intro2abm.com/`)
- NetLogo modeling commons: `http://modelingcommons.org/` is a library of open-source NetLogo models. As with OpenABM, you can search and it can provide a good starting point for research projects.
- Agent-Based Computational Economics, maintained by Leigh Testafon, a professor of economics and math at Iowa State University: `http://www2.econ.iastate.edu/tesfatsi/ace.htm`. This site can be very useful to people interested in economic applications of agent-based modeling.
- The NetLogo team is in the process of developing a new web site, `https://www.netlogo.org/` for the forthcoming release of NetLogo 7.0. The new version of NetLogo is not ready yet, but the website can be interesting to browse, and it links to many useful resources.

Journals

- Journal of Artificial Societies and Social Simulation (`http://jasss.soc.surrey.ac.uk/JASSS.html`) is one of the most important peer-reviewed journals for research on agent-based modeling. It is open-access and web-based, so unlike other journals it is designed primarily to be read online in a browser (although you can also download PDF versions of the articles). Do not be fooled by the open-access or web-based design. It is a very rigorous and well-respected journal.
- Socio-Environmental Systems (`https://sesmo.org/`) covers modeling of combined social and environemntal systems, studying the dynamics of how human societies interact with environmental change, both causing environmental change, and being affected in turn by those changess. Agent-based modeling is an important part of this work.

Computer software

AGENT-BASED MODELING TOOLKITS

- NetLogo: `http://ccl.northwestern.edu/netlogo/`.
  - There is also a version of NetLogo that works in a browser. Its graphics are not quite as nice as the NetLogo software, and the web version is missing some features of the full NetLogo system, but it comes close, and teachers often find it works well to run NetLogo models instead of making students install a software package.
- **NetLogo extensions** (tools to extend and augment NetLogo):
  - Main NetLogo extensions page: `https://github.com/NetLogo/NetLogo/wiki/Extensions`. This has extensions that allow you to read various specialized file formats, interact with other software, such as R and MATLAB, and so forth.
  - BehaviorSearch `http://www.behaviorsearch.org/`. BehaviorSearch is a program that interacts with NetLogo to use genetic algorithms and other techniques to optimize parameters. For instance, if you are trying to figure out what values various parameters of

your NetLogo model make its output best agree with empirical data, or if you are trying to find the values of various parameters that will produce an optimal output (e.g., maximum total wealth for a simulated society), BehaviorSearch can help you. However, you should be aware that BehaviorSearch runs your NetLogo model many times (like tens of thousands to millions of times), so this can be very time consuming.

  - HubNet is a NetLogo extension for participatory modeling, in which many people interact with a single model, often role-playing agents in the model. In a HubNet model, one computer acts as a server and runs the main model, while other computers connect to the server and run individual agents' perspectives on the main model. It is similar to multiplayer games, but the focus is on science, to understand the ways that people interact with each other under different conditions.

- Repast: `https://repast.github.io/`. Repast is a suite of open-source agent-based modeling tools that are much more powerful than NetLogo, but are also more difficult to use. If you go on to do more agent-based modeling after this course and find yourself working on large and complex models for which NetLogo is too limiting, you may want to explore Repast. Repast requires knowledge of either the Java or C++ programming languages. These tools are actively maintained by researchers at Argonne National Laboratory, and include a version for high-performance computing clusters and supercomputers. Repast integrates well with machine-learning systems and has modules for using GIS data for geospatial simulations. There is extensive documentation and tutorial material and a mailing list for support and discussions with other Repast users.

- Mason: `http://cs.gmu.edu/~eclab/projects/mason/` is another powerful agent-based modeling system, using the Java programming language. It integrates well with GIS data, three-dimensional simulation and visualization, and artificial intelligence tools, such as evolutionary algorithms.

OTHER USEFUL SOFTWARE

- R is a very powerful open-source statistical programming language. It can be difficult to learn, but once you learn it, it is very powerful for analyzing data, including large volumes of data generated by BehaviorSpace runs. Even if you don't want to learn how to program in R, you may want to use it in order to use my interactive web-based tool for analyzing BehaviorSpace data. You can download installers for Windows and Mac OS X from `https://cran.rstudio.com`, and if you use Linux, you can install it with `sudo apt install r-base`.

- RStudio provides an integrated development environment for programming in R, which is easier to use for most beginners, and is very powerful for advanced R users. To install it, first install R, and then go to `https://www.rstudio.com/products/rstudio/#Desktop` choose "Download RStudio Desktop Open Source License, and download the appropriate installer for your operating system (Windows or Mac OS X; there are also instructions for installing RStudio on a Linux system).

- NetLogo has extensions that allow it to interface directly with R and Python, so you can run R or Python code from inside a NetLogo model, and you can also run a NetLogo model from inside R and Python programs.

analyzeBehaviorspace

The format of BehaviorSpace output can be very difficult to work with. I have written an interactive application that you can use to convert BehaviorSpace output into more convenient formats. You can access this online at `https://ees4760.jgilligan.org/analyze_behaviorspace/`.

You can also run `analyzeBehaviorspace` on your own computer if you install R (see above). You must have version 3.3 or late of R to use `analyzeBehaviorspace`. After you have installed R on your computer, start R and type the following:

```
install("remotes")
remotes::install_github("jonathan-g/analyzeBehaviorspace")
```

Then you can run `analyzeBehaviorspace` on your computer by starting R and typing the following:

```
library(analyzeBehaviorspace)
launch_abs()
```

The this should open the `analyzeBehaviorspace` app in your web browser. When the app is running, load a `.csv` file containing the table output from a BehaviorSpace run. Then you will see a table of values for all the control and output variables at each tick of each BehaviorSpace run. You can generate summary plots and tables by choosing variables for the x and y axes and a grouping variable (if you choose a grouping variable, each different value of that variable will be represented by a different color in the plots).

The graph will plot the mean value of the y variable for each value of the x variable (grouped by each value of the grouping variable if one is chosen). You can select whether to show points, lines, or both, and whether to just show the mean values of the y variable or to show error bars as well (error bars show plus and minus one standard deviation of the y variable).

If you check the Summary table box, the table will show the mean value and standard deviation of the y variable for each combination of the x and grouping variables.

You can download the plots (as `.png` images) and the summary table (as `.csv` files).

REVISION-CONTROL SOFTWARE

Chapter 5 of the *Agent-Based and Individual-Based Modeling* textbook introduces the idea of version control (also called revision control) on pp. 64–65.

Revision control software is not required for this class, but for people planning a career in research, it can be an essential tool. I recommend that graduate students who plan to use computational methods extensively in their research learn to use one of the popular free/open-source revision-control software packages.

If you plan to use programming (agent based models, other kinds of models, or other computational analysis in which you will write much code), I strongly recommend taking some time to learn a good revision-control tool. If you plan to collaborate with others on programming or modeling research projects, then using revision control will be even more important for sharing your work with each other.

One reason why revision-control software is so important is that if you have a working model and then you go to change something and the model breaks, the revision-control software will let you go back and compare the new (changed) version to the older (working) version, so you can narrow down what you changed that broke the model.

However, it takes a fair bit of time to learn these tools, so there is a trade-off between the time it takes to learn a revision control tool and the benefit you will get from using it over time.

If you plan to use revision control software, I would strongly recommend the free and open source tool, `git`, which has become an industry standard and is very widely supported. Git is available for Windows, Mac OS, and Linux. Often it is installed by default on Linux and Macs. There are clear instructions on checking whether git is already installed and how to install it on all three operating systems at `https://git-scm.com/book/en/v2/Getting-Started-Installing-Git`

A very good free book on how to use Git is *Pro Git* by Scott Chacon and Ben Straub, which you can download from `https://git-scm.com/book`.

If you use Git, I also recommend getting to know the web site `github.com`, which provides a very user-friendly way to publish your code so that other people can use it for their projects. Git runs on your own computer. Github is a site that lets you use git to upload data from your computer in order to share it with the public. Github offers free accounts for anything that you want to share publicly (if you want to use github to store and manage private data, then you need to pay for a private account). In the spirit of making science open and encouraging scientists to build on each other's work, when my research group publishes a computational result, we publish our models and code on github.

Some examples are: https://github.com/JohnNay/datafsm, https://github.com/JohnNay/predMarket/, and https://github.com/JohnNay/forecastVeg.

Git is useful when you are working on your own projects, but if you work collaboratively, git is especially useful for coordinating many people working simultaneously on the same computer code. Indeed, git was written by Linus Torvalds in order to coordinate thousands of programmers simultaneously working on different parts of the Linux operating system.

Revision control can also be very useful for managing your scholarly writing: When I write papers using a plain-text based format (such as $\LaTeX$), I like to use the same revision control software (git, subversion, etc.) that I use for programming projects. By applying revision control to writing, it's easy to check what has changed from one version to another, or to go back and something like, "We massively rewrote section so-and-so in November, but maybe we should go back to the version of that section that we wrote in August."

However, when I work collaboratively, many of my co-authors don't want to learn a fancy software tool, so I just follow the advice that Railsback and Grimm give in Chapter 5: Give each version of the document a new name. When I am doing collaborative writing, each time I send the paper to my co-authors, I make a new version, with the filename following a pattern `Title_2016-01-24-jg.docx`, where "Title" is the working title of the paper. Putting the date in the format `year-month-day` means that when you do a directory listing, the different versions will appear in chronological order. Finally, adding the initials of the author ('jg' for me) who made the most recent edits helps track who did what. Finally, if I send my co-authors more than one new version in a day, I name the second and subsequent versions '`...-jgb.docx`', '`...-jgc.docx`', etc.

Resources on Computation in Social Sciences

Kieran J. Healy, a professor of sociology at Duke University, has written some excellent resources for computational tools in social sciences, including about using R, git, and related tools for analyzing and presenting data.

- The Plain Person's Guide to Plain Text Social Science: http://plain-text.co/. Guidelines for beginning graduate students on useful tools for organizing and managing data, analysis, and writing in quantitative social sciences.
- Data Visualization http://vissoc.co/. Notes from a course Healy taught in Data Visualization for social scientists. Healy turned these notes into a book, *Data Visualization for the Social Sciences*, and posted a draft version of the book online at http://socviz.co/index.html
- Other resources: https://kieranhealy.org/resources/