Collaborating like professionals: integrating NetLogo and GitHub

Nicolas Payette[⋆]

Institute of Cognitive Sciences and Technologies, National Research Council of Italy

Abstract. We are developing an "Online Models" dialogue for NetLogo. It works like NetLogo's "Models Library" dialogue but shows every NetLogo model publicly available on GitHub instead. Our goal is to encourage the sharing of NetLogo models in a way that brings the advantages of version control systems and open-source style collaboration to the world of agent-based modelling.

Keywords: Agent-based modelling · Simulation · NetLogo · Collaboration · Code sharing · Version control · GitHub.

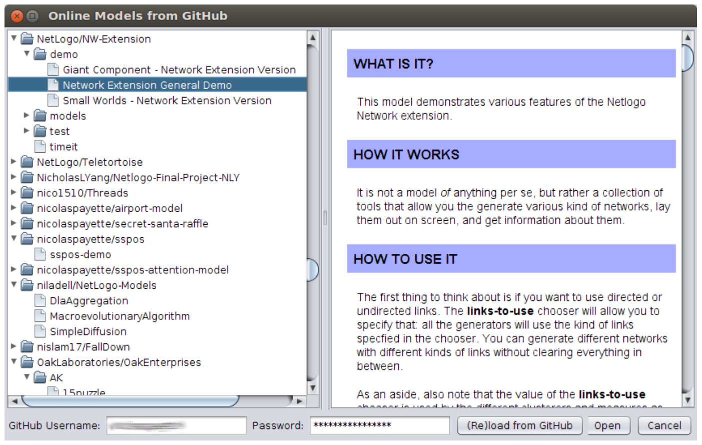
It is tempting to argue that the technical challenges that come with managing complex software artefacts are, by and large, a solved problem. Version control systems have been in common usage since the mid-1980s, and professional programmers learn, sooner or later, that *ad hoc* naming schemes like `myCode-v2.3_temp_Jan2018_(edits).java` are no viable alternative.

Large-scale collaboration also involves, almost by definition, social challenges. While it would be preposterous to claim that these challenges have all been overcome, the open-source software development community can boast of many successful collective endeavours. The Linux operating system and the LibreOffice suite are well-known examples, but a lot of the tools used daily by modellers are also open-source. This is notably the case for the R, Python and Julia programming languages and the Repast [8], GAMA [9,18], MASON [14–16] and NetLogo [19,20] agent-based modelling platforms. All of these, except R, also happen to be hosted on GitHub [3], which provides both version control (through Git) and social collaboration features.

In spite of the benefits, academic researchers (modellers included) have not yet fully embraced the version control/open-source collaboration paradigm. About the reasons for this, we can only (but still will) speculate. Our main goal is to offer a contribution which might help take the agent-based modelling community one step closer to professional-level collaborative software development. For this, we propose to add an "Online Models" dialogue to the NetLogo user interface (fig. 1). It shows every single NetLogo model made publicly available on GitHub. We will suggest future developments of this tool at the end of this paper, but will first look at obstacles to the use of version control systems in the academic

[⋆] Some credit goes to Frank Duncan, with whom these ideas were first discussed, and to Uri Wilensky, without whom NetLogo itself would not exist.

community and at other possible alternatives for sharing and collaborating on NetLogo models.


Online Models from GitHub

- NetLogo/NW-Extension
  - demo
    - Giant Component - Network Extension Version
    - Network Extension General Demo
    - Small Worlds - Network Extension Version
  - models
  - test
  - timeit
- NetLogo/Teletortoise
- NicholasLYang/Netlogo-Final-Project-NLY
- nico1510/Threads
- nicolaspayette/airport-model
- nicolaspayette/secret-santa-raffle
- nicolaspayette/sspos
  - sspos-demo
- nicolaspayette/sspos-attention-model
- niladell/NetLogo-Models
  - DlaAggregation
  - MacroevolutionaryAlgorithm
  - SimpleDiffusion
- nislam17/FallDown
- OakLaboratories/OakEnterprises
  - AK
    - 15puzzle

WHAT IS IT?

This model demonstrates various features of the Netlogo Network extension.

HOW IT WORKS

It is not a model *of* anything per se, but rather a collection of tools that allow you the generate various kind of networks, lay them out on screen, and get information about them.

HOW TO USE IT

The first thing to think about is if you want to use directed or undirected links. The **links-to-use** chooser will allow you to specify that: all the generators will use the kind of links specfied in the chooser. You can generate different networks with different kinds of links without clearing everything in between.

As an aside, also note that the value of the **links-to-use**

GitHub Username: [redacted] Password: ***************

Fig. 1. A screenshot of the “Online Models” dialogue inside NetLogo.

1 Version control systems and obstacles to their use

Simply put, a version control system is “a system that records changes to a file or set of files over time so that you can recall specific versions later.” [7] This is useful for a lone programmer tracking the progress of a personal project, but it borders on essential when multiple programmers are trying to collaborate on a common project. Version control systems track what was changed when by whom. They also offer ways to resolve “merge conflicts” when multiple people tried to change the same thing at the same time.

To fully leverage the power of version control systems, the files dealt with must be plain text files: this allows the user to see exactly what was changed in each file (by herself or someone else) from one version to another and to make decisions accordingly. The need for plain text files for version control to be effective might explain part of the academic community’s reluctance to embrace it: after all, the quintessential academic artefact today is arguably a paper written in Microsoft Word, a piece of software infamous for the inscrutability of its file format[1]. Users of Word somehow manage to collaborate using the “track changes” feature, but their proficiency with this feature is not transferable to something like an agent-based model written in NetLogo.

The learning curve that comes with the use of version control systems might be the single biggest impediment to their adoption. Some graphical user interfaces exist for them, but most version control systems still require occasional trips to

[1] The possibility of using version control systems with LaTeX is one of the advantages it has over Word.

the command line: a land of text and keyboard shortcuts that can seem daunting compared to icons and mouse clicks. The conceptual model associated with some version control systems also requires some getting used to. In Git, for example, the underlying model is a directed acyclic graph of *commits*, with *branches* and *merges*, and changes move from your *working directory* to the *repository* via a *staging area*. These concepts, unknown to most users, need to be added on top of a simple understanding of the computer's file system. Version control also comes with a deeper conceptual shift: you realize that the artefacts that you are dealing with are not just mutating over time, they carry their whole history with them. Once you have made that conceptual shift, the idea of working without version control becomes unbearable.

The NetLogo user experience, by virtue of its "low threshold, high ceiling" philosophy, has some elements in common with a WYSIWYG[2] tool like Word: your model is (usually) a self-contained file that includes its visual interface in addition to its documentation and code. This is both a blessing and a curse. It allows users to immerse themselves in a friendly environment with everything they need at their fingertips, without having to juggle multiple heterogeneous external tools. It does mean, however, that unless they are already familiar and comfortable with them, they tend to forfeit the power of these tools.

One of the purposes of the "Online Models" tool that we are proposing is admittedly to act as a Trojan horse: by making NetLogo models stored on GitHub easily discoverable and loadable from *within* NetLogo, we give users an incentive to integrate version control and collaboration using Git and GitHub through their whole workflow.

2 Alternatives for sharing and collaborating on models

Model description protocols like ODD [10,11] are useful, but not sufficient. Models should not only be described, they should be shared. It is the only way to ensure that results can be reproduced and checked. It is also the best way to increase the chances that other people use and improve the model.

The closest thing to an adopted standard for sharing models in the agent-based modelling community is the OpenABM website of the CoMSES network [1]. At the time of writing, it hosts a library of 484 models with their associated documentation, sometimes giving access to multiple versions of the models. It also offers a peer review service where models are checked for "structural completeness and fulfilling community standards." As long as the sole purpose is to make models publicly available, a site like[3] OpenABM is perfectly adequate.

Our goal, however, should not only be to share models, but also to collaborate on them. The NetLogo Modeling Commons [12, 13] goes many steps beyond simple sharing. Its features are too numerous to be exhaustively listed here, but

[2] "What You See Is What You Get."

[3] The "User Community Models" section of the Center for Connected Learning's website [4] plays a similar role, and non-ABM-specific scientific archiving sites [2,5,6] can also be used for that purpose.

it allows users to modify and comment on each other's models and to (crudely) compare model versions. The fact that it allows models to be uploaded from within NetLogo is also a big advantage.

There are two ways in which the Modeling Commons fall short, though. The first is that model discovery is not integrated into NetLogo. If you want to look at shared models, you have to open your browser, navigate to the website, use their search tools to find an interesting model, download it, save it locally, and then open it in NetLogo. In our "Online Models" tool, by contrast, new models are accessible in just a few clicks or keystrokes, without ever leaving the NetLogo interface. The second downside is that the Modeling Commons only work for NetLogo models. GitHub and Git, by contrast, are language agnostic and bring together most of the open-source development community. It means that models developed on GitHub are potentially discoverable by a lot more people. It also means that the proficiency acquired with Git can be extended to other project artefacts (e.g., analysis code in R, papers in LaTeX, etc.).

3 Future developments

The tool, in its current state of development[4], is pretty straightforward: it's very much like NetLogo's Models Library dialogue, except that the models are stored online on GitHub. A browsable tree appears on the right and the currently selected model's "Info tab" appears on the left (see, again, fig. 1). If you click on the "Open" button, the model is loaded into NetLogo. GitHub users don't need to do anything out of the ordinary for their models to appear in the dialogue: as soon as the model is pushed to a public repository, it shows up in the dialogue.

We believe this sharing/discovery-enabling feature to be sufficient to justify the existence of the tool, but it is the potential for further integration with GitHub that makes it exciting. Here are a few features that could be made easily accessible from within the NetLogo interface:

- Uploading the current model to a new repository on GitHub;
- Committing changes to the current model and pushing them to GitHub;
- "Forking" (i.e., making a modifiable copy of) someone else's model;
- Opening "pull requests" (i.e., proposing changes) for someone else's model;
- Showing the differences between two versions of a model in the code tab.

Taken together, these features would significantly lower the barrier to the adoption of version control for NetLogo models. Combined with the incentive of making a model easily available to all users of our tool, this might help to kickstart the adoption of open collaboration practices for agent-based modellers. In a not so distant future, perhaps our community will also be able to boast of collective achievements on the level of the greatest open-source successes.

[4] The tool is currently being developed in a fork of the NetLogo codebase [17], but the plan is to make it available as a plugin that can easily be installed in the official distribution of NetLogo.

References

1. CoMSES Net, https://www.comses.net/
2. The Dataverse Project, https://dataverse.org/
3. GitHub, https://github.com/
4. NetLogo User Community Models, http://ccl.northwestern.edu/netlogo/models/community/index.cgi
5. Open Science Framework, https://osf.io/
6. Zenodo, https://zenodo.org/
7. Chacon, S., Straub, B.: Pro Git. Apress (Nov 2014)
8. Collier, N.: Repast: An extensible framework for agent simulation. The University of Chicago's Social Science Research **36**, 2003 (2003)
9. Grignard, A., Taillandier, P., Gaudou, B., Vo, D.A., Huynh, N.Q., Drogoul, A.: GAMA 1.6: Advancing the art of complex agent-based modeling and simulation. In: International Conference on Principles and Practice of Multi-Agent Systems. pp. 117–131. Springer (2013)
10. Grimm, V., Berger, U., Bastiansen, F., Eliassen, S., Ginot, V., Giske, J., Goss-Custard, J., Grand, T., Heinz, S.K., Huse, G.: A standard protocol for describing individual-based and agent-based models. Ecological modelling **198**(1-2), 115–126 (2006)
11. Grimm, V., Berger, U., DeAngelis, D.L., Polhill, J.G., Giske, J., Railsback, S.F.: The ODD protocol: a review and first update. Ecological modelling **221**(23), 2760–2768 (2010), http://www.sciencedirect.com/science/article/pii/S030438001000414X
12. Lerner, R., Levy, S.T., Wilensky, U.: Connected Modeling: Design and Analysis of the Modeling Commons. In: Eshet-Alkalai, Y., Caspi, A., Eden, S., Geri, N., Tal-Elhasid, E., Yair, Y. (eds.) Proceedings of the Chais conference on instructional technologies research 2010: Learning in the technological era. pp. 54–60 (2010)
13. Lerner, R.M.: Agent-Based Modeling as a Social Activity. PhD Thesis, Northwestern University (2014)
14. Luke, S., Balan, G.C., Panait, L., Cioffi-Revilla, C., Paus, S.: MASON: A Java multi-agent simulation library. In: Proceedings of Agent 2003 Conference on Challenges in Social Simulation. vol. 9 (2003)
15. Luke, S., Cioffi-Revilla, C., Panait, L., Sullivan, K.: Mason: A new multi-agent simulation toolkit. In: Proceedings of the 2004 swarmfest workshop. vol. 8, pp. 316–327. Michigan, USA (2004)
16. Luke, S., Cioffi-Revilla, C., Panait, L., Sullivan, K., Balan, G.: Mason: A multiagent simulation environment. Simulation **81**(7), 517–527 (2005)
17. Payette, N.: NetLogo Online Models Dialogue (2018), https://github.com/nicolaspayette/NetLogo/tree/online-models
18. Taillandier, P., Vo, D.A., Amouroux, E., Drogoul, A.: GAMA: a simulation platform that integrates geographical information data, agent-based modeling and multi-scale control. In: International Conference on Principles and Practice of Multi-Agent Systems. pp. 242–258. Springer (2010)
19. Tisue, S., Wilensky, U.: Netlogo: A simple environment for modeling complexity. In: International conference on complex systems. vol. 21, pp. 16–21. Boston, MA (2004)
20. Wilensky, U.: NetLogo (1999), http://ccl.northwestern.edu/netlogo/, original-date: 2011-10-17T19:20:01Z