Software, Data and Modelling News

NetLogo meets R: Linking agent-based models with a toolbox for their analysis

Jan C. Thiele [a,*], Volker Grimm [b]

[a] University of Göttingen, Department of Ecoinformatics, Biometrics and Forest Growth, Büsgenweg 4, 37077 Göttingen, Germany
[b] Helmholtz Centre for Environmental Research — UFZ, Department of Ecological Modelling, Permoserstr. 15, 04318 Leipzig, Germany

ARTICLE INFO

ABSTRACT

NetLogo is a software platform for agent-based modelling that is increasingly used in ecological and environmental modelling. So far, for comprehensive analyses of agent-based models (ABMs) implemented in NetLogo, results needed to be written to files and evaluated by using external software, for example R. Ideally, however, it would be possible to call any R function from within a NetLogo program. This would allow sophisticated interactive statistical analysis of model structure and dynamics, using R functions and packages for generating certain statistical distributions and experimental design, and for implementing complex descriptive submodels within ABMs. Here we present an R extension of NetLogo. It consists of only nine new NetLogo primitives for sending data between NetLogo and R and for calling R functions (six additional primitives for debugging). We demonstrate the usage of the R extension with three short examples.

Software availability

Name of the software: NetLogo-R-Extension
Availability: Software and documentation are available at http://netlogo-r-ext.berlios.de
Developers: Jan C. Thiele
Year first available: 2010
Software required: Sun Java (JRE/JDK version 1.5 and higher), NetLogo 4.x, Gnu R (2.6 or higher), rJava package for R
Operation systems: Windows, Linux
Programming language: Java
License: GNU GPL with Linking Exception

1. Introduction

Agent-based models (ABMs) have become an established tool in ecological and environmental modelling (Huse et al., 2002; Porté and Bartelink, 2002; Parker et al., 2003; Bousquet and Le Page, 2004; Grimm and Railsback, 2005). In these models, individual agents, which can be organisms, humans, or institutions, and their behaviour are represented explicitly. ABMs are used when one or more of the following individual-level aspects are considered important for explaining system-level behaviour: heterogeneity among individuals (Uchmanski, 2000), local interactions, and adaptive behaviour which is based on decision making (Grimm, 2008).

Implementing and analysing ABMs can be a challenge because even simple ABMs can generate complex behaviours. For implementing ABMs, various software platforms have been developed (Railsback et al., 2006) which provide specific libraries or programming languages. One of these platforms, NetLogo (Wilensky, 1999), has become very popular in recent years. Originally being more designed for teaching, it is increasingly used for research. It is easy to learn, provides powerful concepts for implementing ABMs, and it has continuously been supported by its developers and a large and growing user community for more than ten years.

NetLogo also provides tools for analysing ABMs. For interactive work, a suite of elements for a graphical user interface exists that allow changes of parameters and settings, visual inspection (Grimm, 2002) of the model world's structure and dynamics, and summary outputs. For batch simulations, the "BehaviorSpace" tool allows simulations to be run repeatedly for different parameter combinations and also allows individual- and system-level output to be written to files. These files are then further analysed with other software, for example Excel or statistics software packages such as R (R Development Core Team, 2008).

Usually, ABM developers using NetLogo would develop and test their model and program using the interactive mode of analysis and

* Corresponding author. Tel.: +49 39 12107; fax: +49 39 3465.
E-mail addresses: jthiele@gwdg.de (J.C. Thiele), volker.grimm@ufz.de (V. Grimm).

only later, once the model is considered good enough to be analysed more thoroughly, would they run simulation experiments via batch mode and use, e.g., R to analyse the results of these experiments.

However, only being able to fully use the statistics and analysis toolbox provided by toolboxes like R in the batch mode can limit model development, testing, and understanding. It would be desirable also to have access to the full toolbox in the interactive mode, so that the direct interaction with the model would include the option of calculating complex summary statistics or other tools for experimental design and analysis provided by R. For example, if we want to understand how model rules and parameters affect the spatial distribution of agents, it would be good to see immediately, without the detour via output files and using external programs, how point pattern statistics such as Ripley's K respond.

Here we present an extension of NetLogo that allows any R function (except functions with multi-line string return values) to be called directly from NetLogo programs (see Fig. 1 and Listing 1). This extension is based on NetLogo's interface (Extension API) for user-defined extensions programmed in Java.

2. New primitives

NetLogo's programming language consists of a large number of commands, or "primitives". Our R extension adds only nine primitives (see documentation and http://netlogo-r-ext.berlios.de/article_resources.php). The new primitives provide means for sending data from NetLogo to R and *vice versa*, for evaluating any R command (with the exception mentioned above) and for observing the processes.

3. Examples

Three examples, which are included in the online attachment (http://netlogo-r-ext.berlios.de/article_resources.php), illustrate how our R extension of NetLogo can be used. The NetLogo program code in the listings contains only the parts where the R extension is used. The complete programs are provided in the examples folder of our R extension of NetLogo.

Listing 1
Sending coordinates of NetLogo agents (turtles) to R and plotting them.

```
extensions [r]
    …
    ;; send turtles variables "who", "xcor" and "ycor" to a new R list with the
    name "agentlist"
    (r:putagent "agentlist" turtles "who" "xcor" "ycor")
    ;; open the plot window of R
    r:eval "Windows()"
    ;; plot the x- and y-positions of the turtles in the R window
    r:eval "plot(agentlist$xcor, agentlist$ycor)"
    …
```

In the first example, (http://netlogo-r-ext.berlios.de/listing2.php) the R extension is used in the *setup* procedure to get random values from a Beta and a Weibull distribution, which are not available in NetLogo. In the *go* procedure the correlation coefficient (Spearman's rho) between the turtle's weight and height variables is calculated and plotted.

The second example makes use of the R package "adehabitat" (http://netlogo-r-ext.berlios.de/listing3.php). Three animals move for 100 time steps and the points visited are stored in a list. These lists are transformed into an R data-frame, which is used for the homerange analysis. The points of vertices of the homerange polygons are sent back to NetLogo and plotted for visualization. In a second step, the "adehabitat" package is used to calculate the size of the homerange area depending on different homerange levels, i. e. the percentage of removed locations for the estimation of the homerange polygon. The higher the level, the lower the removal percentage (see documentation of "adehabitat" package for details). The results are visualized in a NetLogo plot (http://netlogo-r-ext.berlios.de/figure2.php).

In the third example we used the R package "spatstat" to analyse spatial point patterns. It is possible to calculate the L function (based on Ripley's K) for the spatial distribution of the turtles for every time step of the simulation together with the theoretical Poisson function and confidence bands of Complete Spatial Randomness (CSR) from Monte-Carlo-Simulations (http://netlogo-r-ext.berlios.de/listing4.php). In the *setup* procedure, turtles are
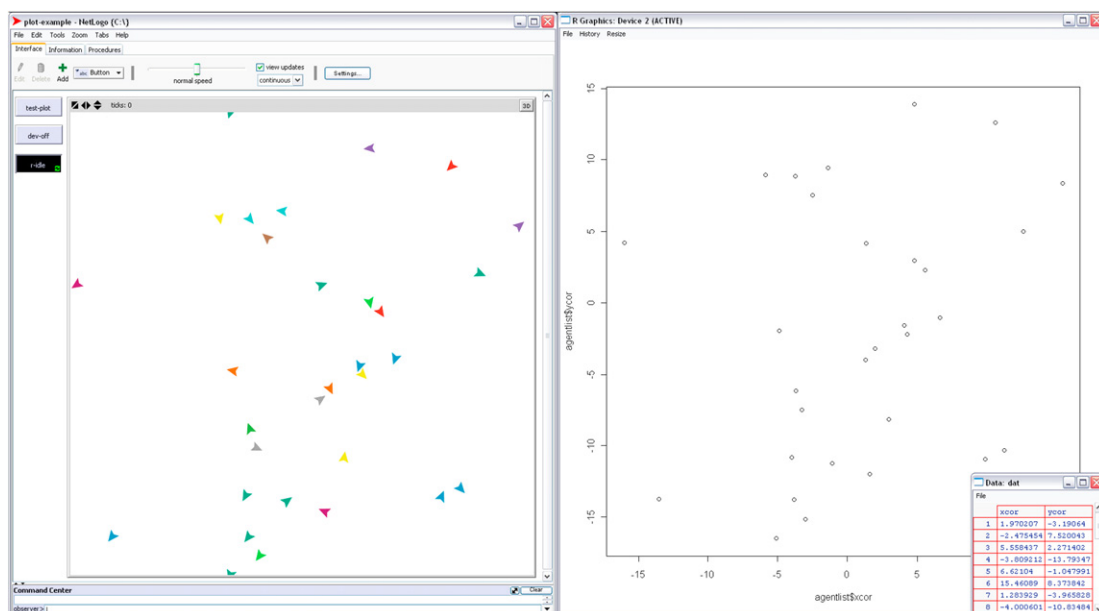

Fig. 1. An example of the interaction between NetLogo and R. Coordinates of NetLogo agents (on the left hand side) were submitted to R and plotted there (on the right) using the new primitive "r:putagent".

created with random positions. In each simulation step (*go* procedure) turtles move around randomly. The new positions are sent to R into a data-frame and are transformed into a point pattern. The L function is then calculated for this point pattern and the results are sent back to NetLogo, where they are transformed and plotted. For illustration purposes we created three extreme scenarios directly in the setup procedure (http://netlogo-r-ext.berlios.de/figure3.php).

4. Concluding remarks

Both NetLogo and R are powerful tools with growing user communities. In the fields of agent-based modelling and statistics, respectively, they are increasingly considered as standard software platforms. Combining these tools to tackle environmental and ecological problems provides many benefits. NetLogo users can utilize the power of R without needing to communicate via data files. This offers new and fascinating opportunities to analyse agent-based models interactively and to implement submodels of, for example, delineating homeranges that are then sensed by the model animals. R users, on the other hand, may be motivated to use NetLogo in cases where R is too limited to implement full-fledged ABMs (Petzoldt and Rinke, 2007).

Our R extension requires users to be familiar with both NetLogo and R, but does not add further complexity. The interface created by our extension consists of only additional nine NetLogo primitives (six additional for debugging), dealing with the communication between NetLogo and R via data and with calling R functions. Our extension will require future updates since the implementation and overall rationale of R and, in particular, NetLogo are continually changing. We will of course attempt to perform these updates ourselves, but since our extension is based on open-source Java code, this process does not depend on us.

References

Bousquet, F., Le Page, C., 2004. Multi-agent simulations and ecosystem management: a review. Ecological Modelling 176, 313–332.
Grimm, V., 2002. Visual debugging: a way of analyzing, understanding, and communicating bottom-up simulation models in ecology. Natural Resource Modeling 15, 23–38.
Grimm, V., 2008. Individual-based models. In: Jørgensen, S.E. (Ed.), Ecological Models. Elsevier, Oxford, pp. 1959–1968.
Grimm, V., Railsback, S.F., 2005. Individual-Based Modeling and Ecology. Princeton University Press, Princeton, N.J.
Huse, G., Giske, J., Salvanes, A.G.V., 2002. Individual-based modelling. In: Hart, P.J.B., Reynolds, J. (Eds.), Handbook of Fish and Fisheries. Blackwell, Oxford, pp. 228–248.
Parker, D.C., Manson, S.M., Janssen, M.A., Hoffmann, M.J., Deadman, P., 2003. Multi-agent systems for the simulation of land-use and land-cover change: a review. Annals of the Association of American Geographers 93, 314–337.
Petzoldt, T., Rinke, K., 2007. Simecol: an object-oriented framework for ecological modeling in R. Journal of Statistical Software 22 (9), 1–31. http://www.jstatsoft.org/v22/i09.
Porté, A., Bartelink, H.H., 2002. Modelling mixed forest growth: a review of models for forest management. Ecological Modelling 150, 141–188.
R Development Core Team, 2008. R: A Language and Environment for Statistical Computing. Available at. R Foundation for Statistical Computing. http://www.r-project.org/.
Railsback, S.F., Lytinen, S.L., Jackson, S.K., 2006. Agent-based simulation platforms: review and development recommendations. Simulation 82, 609–623.
Uchmanski, J., 2000. Individual variability and population regulation: an individual-based model. Oikos 90, 539–548.
Wilensky, U., 1999. NetLogo. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL. http://ccl.northwestern.edu/netlogo.