SoftwareX 7 (2018) 226–233

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

EvoDyn-3s: A Mathematica computable document to analyze evolutionary dynamics in 3-strategy games

Luis R. Izquierdo [a,*], Segismundo S. Izquierdo [b], William H. Sandholm [c]

[a] *Universidad de Burgos, Department of Civil Engineering, Ed. A, Avda. Cantabria s/n, Burgos, 09006, Spain*
[b] *Universidad de Valladolid, Department of Industrial Organization, Paseo del Cauce 59, Valladolid, 47011, Spain*
[c] *University of Wisconsin, Department of Economics, University of Wisconsin, 1180 Observatory Drive, Madison, WI 53706, USA*

ARTICLE INFO

Article history:
Received 23 May 2018
Received in revised form 18 July 2018
Accepted 18 July 2018

Keywords:
Evolutionary dynamics
Game theory
Mathematica
Phase portrait
Stability

ABSTRACT

EvoDyn-3s generates phase portraits of evolutionary dynamics, as well as data for the analysis of their equilibria. The considered evolutionary dynamics are ordinary differential equations based on adaptive processes taking place in a population of players who are randomly and repeatedly matched in couples to play a 2-player symmetric normal-form game with three strategies. *EvoDyn-3s* calculates the rest points of the dynamics using exact arithmetic, and represents them. It also provides the eigenvalues of the Jacobian of the dynamics at the isolated rest points, which are useful to evaluate their local stability. The user only needs to specify the 3×3 payoff matrix of the game and choose the dynamics.

Code metadata

Current code version	v1.0
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX_2018_63
Legal Code License	GNU General Public License (GPL)
Code versioning system used	GitHub
Software code languages, tools, and services used	*Mathematica*
Compilation requirements, operating environments & dependencies	Most *EvoDyn-3s*'s functionality can be used with the free *Wolfram CDF Player*. Full functionality requires *Mathematica*. Both *Wolfram CDF Player* and *Mathematica* run on Windows, Mac and Linux
If available Link to developer documentation/manual	Documentation included within the computable document
Support email for questions	lrizquierdo@ubu.es

Software metadata

Current software version	v1.0
Permanent link to executables of this version	https://github.com/luis-r-izquierdo/EvoDyn-3s/releases/tag/v1.0
Legal Software License	GNU General Public License (GPL)
Computing platforms/Operating Systems	Any that can run *Wolfram CDF* files. E.g. Windows, Mac, Linux, and iOS.
Installation requirements & dependencies	Most *EvoDyn-3s*'s functionality can be used with the free *Wolfram CDF Player*. Full functionality requires *Mathematica*.
If available, link to user manual - if formally published include a reference to the publication in the reference list	Documentation included within the computable document
Support email for questions	lrizquierdo@ubu.es

* Corresponding author.
E-mail addresses: lrizquierdo@ubu.es (L.R. Izquierdo), segis@eii.uva.es (S.S. Izquierdo), whs@ssc.wisc.edu (W.H. Sandholm).
URLs: http://www.luis.izqui.org (L.R. Izquierdo), http://www.segis.izqui.org (S.S. Izquierdo), http://www.ssc.wisc.edu/~whs (W.H. Sandholm).

https://doi.org/10.1016/j.softx.2018.07.006

1. Motivation and significance

Social and biological interactions among agents who may adopt different actions are usually modeled as games. Frequently, these games are studied from an adaptive or evolutionary perspective, leading to systems of ordinary differential equations known as evolutionary game dynamics. A paradigmatic example is the replicator dynamics [1–4], which have become a standard reference case when analyzing adaptive processes.

In general, many of the basic properties of evolutionary game dynamics can be illustrated using games with three strategies [5,6], and applications to the evolution of cooperation are often studied in this framework [7]. To understand and analyze evolutionary game dynamics with three strategies, one of the most useful and intuitive tools are phase portraits, which are geometric representations of the trajectories of the dynamical system in the phase plane. Generating this type of graph typically requires expensive software or customized programming, and this can present a considerable barrier for many researchers studying evolutionary processes. *EvoDyn-3s* has been designed to overcome this barrier.

Specifically, *EvoDyn-3s* provides high-quality print-ready vector phase portraits for a diverse group of evolutionary game dynamics with three strategies. Using it does not require any programming and most of its functionality can be run with the free *Wolfram CDF Player*.[1] *EvoDyn-3s* also calculates and presents the equilibria of the selected dynamics and performs an eigenvalue analysis of the linearized dynamics *using exact arithmetic*, a feature that is not available in other programs and can be very useful for theoretical analysis.

The software most closely related to *EvoDyn-3s* is *Dynamo* [8,9]. *Dynamo* is also open-source software that runs on *Mathematica*, and has also been designed to create phase diagrams and other images related to dynamical systems from evolutionary game theory. *Dynamo* can be used to generate graphs for single-population games with 3 or 4 strategies and for some multipopulation games. *Dynamo* is more flexible and general than *EvoDyn-3s*, but significantly less user-friendly, and it uses numerical approximations rather than exact arithmetic. Another software somewhat related to *EvoDyn-3s* is *PDToolbox* [10], which is a set of functions coded in *Matlab* for analyzing some evolutionary dynamics, as well as finite-population agent-based models related to those dynamics. The Python package *egtplot* [11] creates phase diagrams for the replicator dynamics. While narrower in scope and less user-friendly than *EvoDyn-3s*, *egtplot* has the commendable feature of running on an open-source platform. Lastly, *ABED* [12] (**A**gent-**B**ased **E**volutionary **D**ynamics) is also free and completely open-source software for simulating adaptive processes, but in finite populations. It provides a complementary approach to the analysis of evolutionary dynamics followed in *EvoDyn-3s*, in the sense that many of the adaptive processes considered and implemented in *ABED* (following an agent-based approach) can be approximated –for sufficiently large populations – by differential equations corresponding to the evolutionary dynamics implemented in *EvoDyn-3s*.

2. Software description

EvoDyn-3s is a computable document written in *Mathematica* language. The document contains the executable program, detailed instructions on how to use it, and the source code. There is no need to compile the code. The program can be used directly, by simply opening the computable document with the free *Wolfram CDF Player* or with *Mathematica*.

Fig. 1 shows the interface of *EvoDyn-3s*. The left part contains a series of input boxes and various controls that are used to set the values of all parameters. The right part shows the main output of the program: a phase portrait in the 2-dimensional simplex, a table showing all the isolated rest points and the eigenvalues of the Jacobian of the chosen dynamic at each of the isolated rest points (if the Jacobian is defined), and another table showing the components of rest points (if any exist).

The effect of changing the value of any parameter (except the payoff matrix) on the phase portrait and on the computation of rest points and eigenvalues is immediate, i.e. there is no need to compile or rerun the program.[2] Thus, for example, the user can gradually move any parameter slider and immediately appreciate how this change affects the output of the program.

2.1. Software architecture

EvoDyn-3s conducts the following high-level operations, which are sketched in Fig. 2:

- Creates the system of differential equations using the following input provided by the user: payoff matrix, baseline dynamic, probability of random strategy μ, and –only for dynamics Logit and Single-match imitative logit – parameter η. The generated system of differential equations is of the form $(\dot{x}_1, \dot{x}_2, \dot{x}_3) = f(x_1, x_2, x_3)$, where x_i represents the fraction of the population using strategy i.
- Solves the system of differential equations numerically for various initial conditions and represents the solutions. The number of initial conditions and the length of the computed trajectories can be controlled by the user.
- Represents a series of orbits and arrows showing the direction of movement in the 2-dimensional simplex $\{(x_1, x_2, x_3) \mid \sum_{i=1}^{3} x_i = 1 \text{ and } x_i \geq 0\}$.
- Colors the background of the simplex according to the speed of the dynamic, using the color gradient selected by the user.
- Computes the rest points of the dynamic using exact arithmetic.
- Computes numerical approximations to the isolated rest points. Shows them in a table and represents them in the 2-dimensional simplex.
- For dynamics where the Jacobian is defined, computes the eigenvalues of the Jacobian of the dynamic at the exact isolated rest points using exact arithmetic. Computes numerical approximations to the eigenvalues and shows them in a table.
- Shows the components of rest points in a table and represents them in the 2-dimensional simplex.

2.2. Software functionalities

EvoDyn-3s generates phase portraits of evolutionary game dynamics, colors the background according to their speed (i.e. the modulus of the derivative vector), calculates the rest points, and provides the eigenvalues of the Jacobian at isolated rest points in order to analyze their local stability (in dynamics where the Jacobian is defined).

The user can analyze any 3×3 game by setting the values a_{ij} of the 3×3 payoff matrix, which represent the payoff that a player using strategy $i \in \{1, 2, 3\}$ obtains when interacting with a player using strategy $j \in \{1, 2, 3\}$. There is also a list of predefined games that the user can choose from.

[1] *EvoDyn-3s*'s full functionality requires *Mathematica*.

[2] The payoff matrix is not updated automatically, but only when the user clicks on the button "update". This is a purposeful implementation, since the user often wants to change several payoff values at the same time.